Luck Matters: Understanding Training Dynamics of Deep ReLU Networks

Yuandong Tian **Tina Jiang** **Qucheng Gong** **Ari Morcos**
Facebook AI Research
{yuandong, tinayujiang, qucheng, arimorcos}@fb.com

Abstract

We analyze the dynamics of training deep ReLU networks and their implications on generalization capability. Using a teacher-student setting, we discovered a novel relationship between the gradient received by hidden student nodes and the activations of teacher nodes for deep ReLU networks. With this relationship and the assumption of small overlapping teacher node activations, we prove that (1) student nodes whose weights are initialized to be close to teacher nodes converge to them at a faster rate, and (2) in over-parameterized regimes and 2-layer case, while a small set of lucky nodes do converge to the teacher nodes, the fan-out weights of other nodes converge to zero. This framework provides insight into multiple puzzling phenomena in deep learning like over-parameterization, implicit regularization, lottery tickets, etc. We verify our assumption by showing that the majority of BatchNorm biases of pre-trained VGG11/13/16/19 models are negative. Experiments on (1) random deep teacher networks with Gaussian inputs, (2) teacher network pre-trained on CIFAR-10 and (3) extensive ablation studies validate our multiple theoretical predictions. Code is available at `https://github.com/facebookresearch/luckmatters`.

1 Introduction

Although neural networks have made strong empirical progress in a diverse set of domains (e.g., computer vision [17, 33, 11], speech recognition [12, 1], natural language processing [23, 4], and games [31, 32, 36, 24]), a number of fundamental questions still remain unsolved. How can Stochastic Gradient Descent (SGD) find good solutions to a complicated non-convex optimization problem? Why do neural networks generalize? How can networks trained with SGD fit both random noise and structured data [39, 18, 25], but prioritize structured models, even in the presence of massive noise [28]? Why are flat minima related to good generalization? Why does over-parameterization lead to better generalization [26, 40, 34, 27, 20]? Why do lottery tickets exist [7, 8]?

In this paper, we propose a theoretical framework for multilayered ReLU networks. Based on this framework, we try to explain these puzzling empirical phenomena with a unified view. We adopt a teacher-student setting where the label provided to an *over-parameterized* deep student ReLU network is the output of a fixed teacher ReLU network of the same depth and unknown weights (Fig. 1(a)). Here over-parameterization means that at each layer, the number of nodes in student network is more than the number of nodes in the teacher network. In this perspective, hidden student nodes are randomly initialized with different activation regions (Fig. 2(a)). During optimization, student nodes compete with each other to explain teacher nodes. From this setting, Theorem 4 shows that *lucky* student nodes which have greater overlap with teacher nodes converge to those teacher nodes at a *fast rate*, resulting in *winner-take-all* behavior. Furthermore, Theorem 5 shows that in the 2-layer case, if a subset of student nodes are close to the teachers', they converge to them and the fan-out weights of other irrelevant nodes of the same layer vanishes.

Preprint. Work in progress.

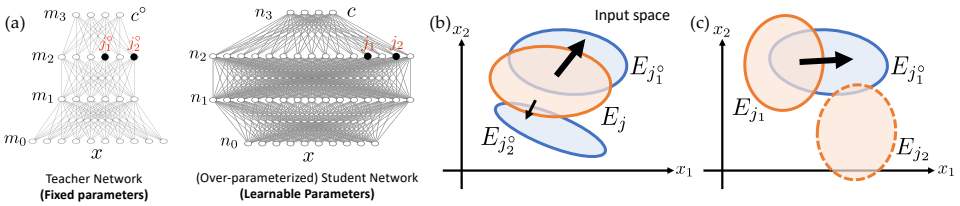


Figure 1: **(a)** Teacher-Student Setting. For each node j, the activation region is $E_j = \{x : f_j(x) > 0\}$. **(b)** node j initialized to overlap substantially with a teacher node j_1° converges faster towards j_1° (Thm. 4). **(c)** Student nodes initialized to be close to teacher converges to them, while the fan-out weights of other irrelevant student nodes goes to zero. (Thm. 5).

With this framework, we try to intuitively explain various neural network behaviors as follows:

Fitting both structured and random data. Under gradient descent dynamics, some student nodes, which happen to overlap substantially with teacher nodes, will move into the teacher node and cover them. This is true for both structured data that corresponds to small teacher networks with few intermediate nodes, or noisy/random data that correspond to large teachers with many intermediate nodes. This explains why the same network can fit both structured and random data (Fig. 2(a-b)).

Over-parameterization. In over-parameterization, lots of student nodes are initialized randomly at each layer. Any teacher node is more likely to have a substantial overlap with some student nodes, which leads to fast convergence (Fig. 2(a) and (c), Thm. 4), consistent with [7, 8]. This also explains that training models whose capacity just fit the data (or teacher) yields worse performance [20].

Flat minima. Deep networks often converge to "flat minima" whose Hessian has a lot of small eigenvalues [29, 30, 22, 3]. Furthermore, while controversial [5], flat minima seem to be associated with good generalization, while sharp minima often lead to poor generalization [13, 15, 37, 21]. In our theory, when fitting with structured data, only a few lucky student nodes converge to the teacher, while for other nodes, their fan-out weights shrink towards zero, making them (and their fan-in weights) irrelevant to the final outcome (Thm. 5), yielding flat minima in which movement along most dimensions ("unlucky nodes") results in minimal change in output. On the other hand, sharp minima is related to noisy data (Fig. 2(d)), in which more student nodes match with the teacher.

Implicit regularization. On the other hand, the snapping behavior enforces *winner-take-all*: after optimization, a teacher node is fully covered (explained) by a few student nodes, rather than splitting amongst student nodes due to over-parameterization. This explains why the same network, once trained with structured data, can generalize to the test set.

Lottery Tickets. Lottery Tickets [7, 8, 41] is an interesting phenomenon: if we reset "salient weights" (trained weights with large magnitude) back to the values before optimization but after initialization, prune other weights (often $> 90\%$ of total weights) and retrain the model, the test performance is the same or better; if we reinitialize salient weights, the test performance is much worse. In our theory, the salient weights are those lucky regions (E_{j_3} and E_{j_4} in Fig. 3) that happen to overlap with some teacher nodes after initialization and converge to them in optimization. Therefore, if we reset their weights and prune others away, they can still converge to the same set of teacher nodes, and potentially achieve better performance due to less interference with other irrelevant nodes. However, if we reinitialize them, they are likely to fall into unfavorable regions which cannot cover teacher nodes, and therefore lead to poor performance (Fig. 3(c)), just like in the case of under-parameterization. Recently, Supermask [41] shows that a supermask can be found from winning tickets. If it is applied to initialized weights, the network without training gives much better test performance than chance. This is also consistent with the intuitive picture in Fig. 3(b).

2 Mathematical Framework

Notation. Consider a student network and its associated teacher network (Fig. 1(a)). Denote the input as x. For each node j, denote $f_j(x)$ as the activation, $f'_j(x)$ as the ReLU gating (for the top-layer, $f'_c(x)$ and $f'_{c^\circ}(x)$ are always 1), and $g_j(x)$ as the backpropagated gradient, all as functions of x. We use the superscript $^\circ$ to represent a teacher node (e.g., j°). Therefore, g_{j° never appears as teacher nodes are not updated. We use w_{jk} to represent weight between node j and k in the student network. Similarly, $w^*_{j^\circ k^\circ}$ represents the weight between node j° and k° in the teacher network.

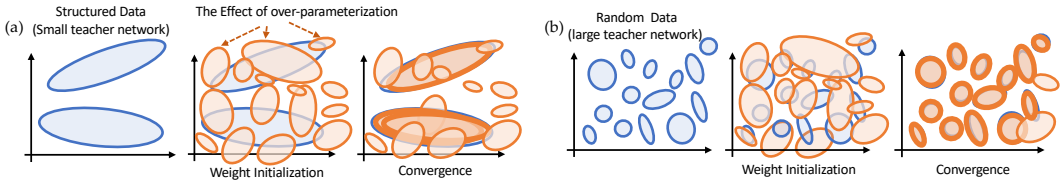


Figure 2: Explanation of implicit regularization. Blue are activation regions of teacher nodes, while orange are students'. **(a)** When the data labels are structured, the underlying teacher network is small and each layer has few nodes. Over-parameterization (lots of red regions) covers them all. Moreover, those student nodes that heavily overlap with the teacher nodes converge faster (Thm. 4), yield good generalization performance. **(b)** If a dataset contains random labels, the underlying teacher network that can fit to it has a lot of nodes. Over-parameterization can still handle them and achieves zero training error.

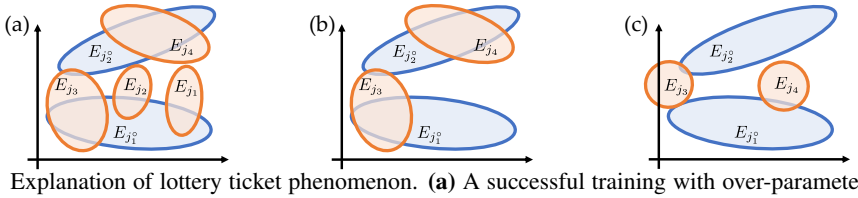


Figure 3: Explanation of lottery ticket phenomenon. **(a)** A successful training with over-parameterization (2 filters in the teacher network and 4 filters in the student network). Node j_3 and j_4 are lucky draws with strong overlap with two teacher node j_1° and j_2°, and thus converges with high weight magnitude. **(b)** Lottery ticket phenomenon: initialize node j_3 and j_4 with the same initial weight, clamp the weight of j_1 and j_2 to zero, and retrain the model, the test performance becomes better since j_3 and j_4 still converge to their teacher node, respectively. **(c)** If we reinitialize node j_3 and j_4, it is highly likely that they are not overlapping with teacher node j_1° and j_2° so the performance is not good.

We focus on multi-layered ReLU networks. We use the following equality extensively: $\sigma(x) = \sigma'(x)x$. For ReLU node j, we use $E_j \equiv \{x : f_j(x) > 0\}$ as the activation region of node j.

Teacher network versus Dataset. The reason why we formulate the problem using teacher network rather than a dataset is the following: (1) It leads to a nice and symmetric formulation for multi-layered ReLU networks (Thm. 1). (2) A teacher network corresponds to an infinite size dataset, which separates the finite sample issues from induction bias in the dataset, which corresponds to the structure of teacher network. (3) If student weights can be shown to converge to teacher ones, generalization bound can naturally follow for the student. (4) The label complexity of data generated from a teacher is automatically reduced, which could lead to better generalization bound. On the other hand, a bound for arbitrary function class can be hard.

Objective. We assume that both the teacher and the student output probabilities over C classes. We use the output of teacher as the input of the student. At the top layer, each node c in the student corresponds to each node c° in the teacher. Therefore, the objective is:

$$\min_{\mathbf{w}} J(\mathbf{w}) = \frac{1}{2}\mathbb{E}_x \left[\|f_c(x) - f_{c^\circ}(x)\|^2\right] \tag{1}$$

By the backpropagation rule, we know that for each sample x, the (negative) gradient $g_c(x) \equiv \partial J/\partial f_c = f_{c^\circ}(x) - f_c(x)$. The gradient gets backpropagated until the first layer is reached.

Note that here, the gradient $g_c(x)$ sent to node c is *correlated* with the activation of the corresponding teacher node $f_{c^\circ}(x)$ and other student nodes at the same layer. Intuitively, this means that the gradient "pushes" the student node c to align with class c° of the teacher. If so, then the student learns the corresponding class well. A natural question arises:

Are student nodes at intermediate layers correlated with teacher nodes at the same layers?

One might wonder this is hard since the student's intermediate layer receives no *direct supervision* from the corresponding teacher layer, but relies only on backpropagated gradient. Surprisingly, the following theorem shows that it is possible for every intermediate layer:

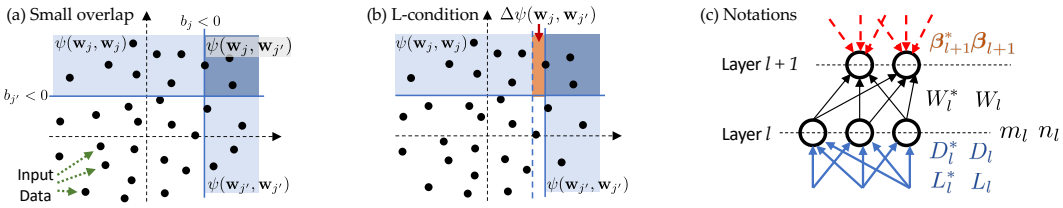


Figure 4: **(a)** Small overlaps between node activations. Figure drawn in the space spanned by the activations of last layer so all decision boundaries are linear. **(b)** Lipschitz condition (Assumption 2). **(c)** Notation in Thm. 2.

Theorem 1 (Recursive Gradient Rule)**.** *If all nodes j at layer l satisfies Eqn. 2*

$$g_j(x) = f'_j(x)\left[\sum_{j^\circ}\beta^*_{jj^\circ}(x)f_{j^\circ}(x) - \sum_{j'}\beta_{jj'}(x)f_{j'}(x)\right], \tag{2}$$

then all nodes k at layer $l-1$ also satisfies Eqn. 2 with $\beta^*_{kk^\circ}(x)$ and $\beta_{kk'}(x)$ defined recursively from top to bottom-layer as follows:

$$\beta^*_{kk^\circ}(x) \equiv \sum_{jj^\circ} w_{jk}f'_j(x)\beta^*_{jj^\circ}(x)f'_{j^\circ}(x)w^*_{j^\circ k^\circ}, \quad \beta_{kk'}(x) \equiv \sum_{jj'} w_{jk}f'_j(x)\beta_{jj'}(x)f'_{j'}(x)w_{j'k'} \tag{3}$$

And for the base case where node c and c° are at the top-most layer, $\beta^*_{cc^\circ}(x) = \beta_{cc^\circ}(x) = \delta(c-c^\circ)$ (i.e., 1 when c corresponds to c°, and 0 otherwise).

Note that Theorem 1 applies to arbitrarily deep ReLU networks and allows different number of nodes for the teacher and student. The role played by ReLU activation is to make the expression of β concise, otherwise β_{kk° and $\beta^*_{kk^\circ}$ can take a very complicated (and asymmetric) form.

In particular, we consider the *over-parameterization* setting: the number of nodes on the student side is much larger (e.g., 5-10x) than the number of nodes on the teacher side. Using Theorem 1, we discover a novel and concise form of gradient update rule:

Assumption 1 (Separation of Expectations)**.**

$$\mathbb{E}_x\left[\beta^*_{jj^\circ}(x)f'_j(x)f'_{j^\circ}(x)f_k(x)f_{k^\circ}(x)\right] = \mathbb{E}_x\left[\beta^*_{jj^\circ}(x)\right]\mathbb{E}_x\left[f'_j(x)f'_{j^\circ}(x)\right]\mathbb{E}_x\left[f_k(x)f_{k^\circ}(x)\right] \tag{4}$$

$$\mathbb{E}_x\left[\beta_{jj'}(x)f'_j(x)f'_{j'}(x)f_k(x)f_{k'}(x)\right] = \mathbb{E}_x\left[\beta_{jj'}(x)\right]\mathbb{E}_x\left[f'_j(x)f'_{j'}(x)\right]\mathbb{E}_x\left[f_k(x)f_{k'}(x)\right] \tag{5}$$

Theorem 2. *If Assumption 1 holds, the gradient dynamics of deep ReLU networks with objective (Eqn. 1) is:*

$$\dot{W}_l = L^*_l W^*_l H^*_{l+1} - L_l W_l H_{l+1} \tag{6}$$

Here we explain the notations. $W^*_l = \left[\mathbf{w}^*_1, \ldots, \mathbf{w}^*_{m_l}\right]$ is m_l teacher weights, $\boldsymbol{\beta}^*_{l+1} = \mathbb{E}_x\left[\beta_{jj^\circ}(x)\right]$, $d^*_{jj^\circ} = \mathbb{E}_x\left[f'_j(x)f'_{j^\circ}(x)\right]$ and $D^*_l = [d^*_{jj^\circ}]$, $H^*_{l+1} = [h_{jj^\circ}] = \boldsymbol{\beta}^*_{l+1} \circ D_l$, $l^*_{jj^\circ} = \mathbb{E}_x\left[f_j(x)f_{j^\circ}(x)\right]$ and $L^*_l = [l^*_{jj^\circ}]$. We can define similar notations for W (which has n_l columns/filters), $\boldsymbol{\beta}$, D, H and L (Fig. 4(c)). At the lowest layer $l = 0$, $L_0 = L^*_0$, at the highest layer $l = l_{\max} - 1$ where there is no ReLU, we have $\boldsymbol{\beta}_{l_{\max}} = \boldsymbol{\beta}^*_{l_{\max}} = H_{l_{\max}} = H^*_{l_{\max}} = I$ due to Eqn. 1. According to network structure, $\boldsymbol{\beta}_{l+1}$ and $\boldsymbol{\beta}^*_{l+1}$ only depends on weights $W_{l+1}, \ldots, W_{l_{\max}-1}$, while L_l and L^*_l only depend on $W_0, \ldots, W_{l-1}$.

3 Analysis on the Dynamics

In the following, we will use Eqn. 6 to analyze the dynamics of the multi-layer ReLU networks. For convenience, we first define the two functions ψ_l and ψ_d (σ is the ReLU function):

$$\psi_l(\mathbf{w}, \mathbf{w}') = \mathbb{E}_x\left[\sigma(\mathbf{w}^T x)\sigma(\mathbf{w}'^T x)\right], \quad \psi_d(\mathbf{w}, \mathbf{w}') = \mathbb{E}_x\left[\mathbb{I}(\mathbf{w}^T x)\mathbb{I}(\mathbf{w}'^T x)\right]. \tag{7}$$

We assume these two functions have the following property .

Assumption 2 (Lipschitz condition)**.** *There exists K_d and K_l so that:*

$$\|\psi_i(\mathbf{w}, \mathbf{w}_1) - \psi_i(\mathbf{w}, \mathbf{w}_2)\| \le \psi_i(\mathbf{w}, \mathbf{w}_1)(1 + K_i\|\mathbf{w}_1 - \mathbf{w}_2\|), \quad i \in \{d, l\} \tag{8}$$

Using this, we know that $d_{jj'} = \psi_d(\mathbf{w}_j, \mathbf{w}_{j'})$, $d^*_{jj'} = \psi_d(\mathbf{w}_j, \mathbf{w}^*_{j'})$, and so on. For brevity, denote $d^{**}_{jj'} = \psi_d(\mathbf{w}^*_j, \mathbf{w}^*_{j'})$ (when notation j°_1 is heavy) and so on. We impose the following assumption:

Assumption 3 (Small Overlap between teacher nodes)**.** *There exists $\epsilon_l \ll 1$ and $\epsilon_d \ll 1$ so that:*

$$d^{**}_{j_1 j_2} \leq \epsilon_d d^{**}_{j_1 j_1} \text{ (or } \epsilon_d d^{**}_{j_2 j_2}), \quad l^{**}_{j_1 j_2} \leq \epsilon_l l^{**}_{j_1 j_1} \text{ (or } \epsilon_l l^{**}_{j_2 j_2}), \quad \text{for } j_1 \neq j_2 \tag{9}$$

Intuitively, this means that the probability of the simultaneous activation of two teacher nodes j_1 and j_2 is small. If we have sufficient training data to cover the input space, then a sufficient condition for Assumption 3 to happen is that the teacher has negative bias, which means that they *cut corners* in the space spanned by the node activations of the lower layer (Fig. 4a). We have empirically verified that the majority of biases in BatchNorm layers (after the data are whitened) are negative in VGG11/16 trained on ImageNet (Sec. 4.1).

3.1 Effects of BatchNorm

Batch Normalization [14] has been extensively used to speed up the training, reduce the tuning efforts and improve the test performance of neural networks. Here we use an interesting property of BatchNorm: the total "energy" of the incoming weights of each node j is conserved over training iterations:

Theorem 3 (Conserved Quantity in Batch Normalization)**.** *For Linear $\rightarrow$ ReLU $\rightarrow$ BN or Linear $\rightarrow$ BN $\rightarrow$ ReLU configuration, $\|\mathbf{w}_j\|$ of a filter j before BN remains constant in training. (Fig. 15).*

See Appendix for the proof. The similar lemma is also in [2]. This may partially explain why BN has stabilization effect: energy will not leak from one layer to nearby ones. Due to this property, in the following, for convenience we assume $\|\mathbf{w}_j\|^2 = \|\mathbf{w}^*_j\|^2 = 1$, and the gradient $\dot{\mathbf{w}}_j$ is always orthogonal to the current weight $\mathbf{w}_j$. Note that on the teacher side we can always push the magnitude component to the upper layer; on the student side, random initialization naturally leads to constant magnitude of weights.

3.2 Same number of student nodes as teacher

We start with a simple case first. Consider that we only analyze layer l without over-parameterization, i.e., $n_l = m_l$. We also assume that $L^*_l = L_l = I$, i.e., the input of that layer l is whitened, and the top-layer signal is uniform, i.e., $\boldsymbol{\beta}^*_{l+1} = \boldsymbol{\beta}_{l+1} = \mathbf{1}\mathbf{1}^T$ (all β entries are 1). Then the following theorem shows that weight recovery could follow (we use j' as j°).

Theorem 4. *For dynamics $\dot{\mathbf{w}}_j = P^\perp_{\mathbf{w}_j}(W^*\mathbf{h}^*_j - W\mathbf{h}_j)$, where $P^\perp_{\mathbf{w}_j} \equiv I - \mathbf{w}_j\mathbf{w}_j^T$ is a projection matrix into the orthogonal complement of $\mathbf{w}_j$. $\mathbf{h}^*_j$, $\mathbf{h}_j$ are corresponding j-th column in H^* and H. Denote $\theta_j = \angle(\mathbf{w}_j, \mathbf{w}^*_j)$ and assume $\theta_j \leq \theta_0$. If $\gamma = \cos\theta_0 - (m-1)\epsilon_d M_d > 0$, then $\mathbf{w}_j \rightarrow \mathbf{w}^*_j$ with the rate $1 - \eta\bar{d}\gamma$ (η is learning rate). Here $\bar{d} = [1 + 2K_d \sin(\theta_0/2)] \min_j d^{*0}_{jj}$ and $M_d = (1 + K_d)\left[1 + 2K_d \sin(\theta_0/2)\right]^2 / \cos\frac{\theta_0}{2}$.*

See Appendix for the proof. Here we list a few remarks:

Faster convergence near $\mathbf{w}^*_j$. we can see that due to the fact that h^*_{jj} in general becomes larger when $\mathbf{w}_j \rightarrow \mathbf{w}^*_j$ (since $\cos\theta_0$ can be close to 1), we expect a *super-linear* convergence near $\mathbf{w}^*_j$. This brings about an interesting *winner-take-all* mechanism: if the initial overlap between a student node j and a particular teacher node is large, then the student node will snap to it (Fig. 1(c)).

Importance of projection operator $P^\perp_{\mathbf{w}_j}$. Intuitively, the projection is needed to remove any ambiguity related to weight scaling, in which the output remains constant if the top-layer weights are multiplied by a constant α, while the low-layer weights are divided by α. Previous works [6] also uses similar techniques while we justify it with BN. Without $P^\perp_{\mathbf{w}_j}$, convergence can be harder.

Top-down modulation. Note that here we assume the top-layer signal is uniform, which means that according to $\beta^*_{kk^\circ}$, there is no preference on which student node k corresponds to which teacher node k°. If there is a preference (e.g., $\beta^*_{kk^\circ} = \delta(k - k^\circ)$), then from the proof, the cross-term $h^*_{kk^\circ}$ will be suppressed due to $\beta^*_{kk^\circ}$, making convergence easier. As we will see next, such a *top-down modulation* plays an important role for 2-layer and over-parameterization case. We believe that it also plays a similar role for deep networks.

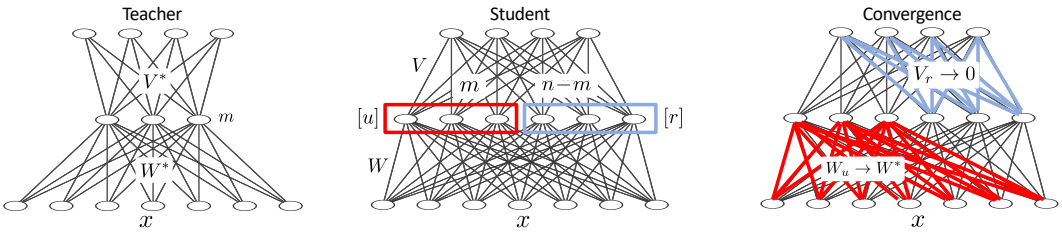


Figure 5: Over-parameterization and top-down modulation. Thm. 5 shows that under certain conditions, the relevant weights $W_u \to W^*$ and weights connecting to irrelevant student nodes $V_r \to 0$.

3.3 Over-Parameterization and Top-down Modulation in 2-layer Network

In the over-parameterization case ($n_l > m_l$, e.g., 5-10x), we arrange the variables into two parts: $W = [W_u, W_r]$, where W_u contains m_l columns (same size as W^*), while W_r contains $n_l - m_l$ columns. We use $[u]$ (or u-set) to specify nodes $1 \le j \le m$, and $[r]$ (or r-set) for the remaining part.

In this case, if we want to show "the main component" W_u converges to W^*, we will meet with one core question: to where will W_r converge, or whether W_r will even converge at all? We need to consider not only the dynamics of the current layer, but also the dynamics of the upper layer. Using a 1-hidden layer over-parameterized ReLU network as an example, Theorem 5 shows that the upper-layer dynamics $\dot{V} = L^* V^* - LV$ automatically apply *top-down modulation* to suppress the influence of W_r, regardless of their convergence. Here $V = \begin{bmatrix} V_u \\ V_r \end{bmatrix}$, where V_u are the weight components of u-set. See Fig. 5.

Theorem 5 (Over-Parameterization and Top-down Modulation)**.** *Consider $\dot{W} = W^* H^* - WH$ with over-parameterization ($n > m$) and its upper-layer dynamics $\dot{V} = L^* V^* - LV$. Assume that initial value W_u^0 is close to W^*: $\theta_j = \angle(\mathbf{w}_j, \mathbf{w}_j^*) \le \theta_0$ for $j \in [u]$. If (1) Assumption 3 holds for all pairwise combination of columns of W^* **and** W_r^0, and (2) there exists $\gamma = \gamma(\theta_0, m) > 0$ and $\bar{\lambda}$ so that Eqn. 41 and Eqn. 42 holds, then $W_u \to W^*$, $V_u \to V^*$ and $V_r \to 0$ with rate $1 - \eta\bar{\lambda}\gamma$.*

See Appendix for the proof (and definition of $\bar{\lambda}$ in Eqn. 45). The intuition is: if W_u is close to W^* and W_r are far away from them due to Assumption 3, the off-diagonal elements of L and L^* are smaller than diagonal ones. This causes V_u to move towards V^* and V_r to move towards zero. When V_r becomes small, so does $\beta_{jj'}$ for $j \in [r]$ or $j' \in [r]$. This in turn suppresses the effect of W_r and accelerates the convergence of W_u. $V_r \to 0$ exponentially so that W_r stays close to its initial locations, and Assumption 3 holds for all iterations. A few remarks:

Flat minima. Since $V_r \to 0$, W_r can be changed arbitrarily without affecting the outputs of the neural network. This could explain why there are many flat directions in trained networks, and why many eigenvalues of the Hessian are close to zero [29].

Understanding of pruning methods. Theorem 5 naturally relates two different unstructured network pruning approaches: pruning small weights in magnitude [9, 7] and pruning weights suggested by Hessian [19, 10]. It also suggests a principled structured pruning method: instead of pruning a filter by checking its weight norm, pruning accordingly to its top-down modulation.

Accelerated convergence and learning rate schedule. For simplicity, the theorem uses a uniform (and conservative) γ throughout the iterations. In practice, γ is initially small (due to noise introduced by r-set) but will be large after a few iterations when V_r vanishes. Given the same learning rate, this leads to accelerated convergence. At some point, the learning rate η becomes too large, leading to fluctuation. In this case, η needs to be reduced.

Many-to-one mapping. Theorem 5 shows that under strict conditions, there is one-to-one correspondence between teacher and student nodes. In general this is not the case. Two students nodes can be both in the vicinity of a teacher node $\mathbf{w}_j^*$ and converge towards it, until that node is fully explained. We leave it to the future work for rigid mathematical analysis of many-to-one mappings.

Random initialization. One nice thing about Theorem 5 is that it only requires the initial $\|W_u - W^*\|$ to be small. In contrast, there is *no* requirement for small $\|V_r\|$. Therefore, we could expect that with more over-parameterization and random initialization, in each layer l, it is more likely to find the u-set (of fixed size m_l), or the *lucky weights*, so that W_u is quite close to W^*. At the same

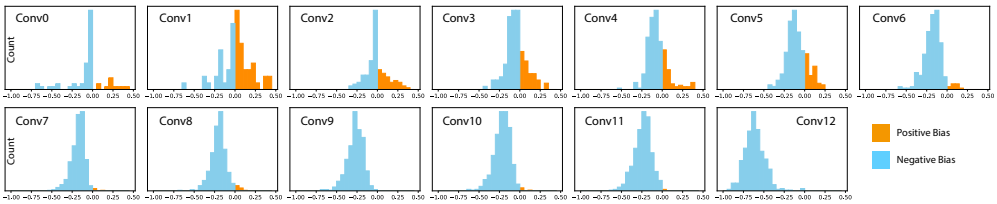


Figure 6: Distribution of BatchNorm bias in pre-trained VGG16 on ImageNet. Orange/blue are positive/negative biases. `Conv0` corresponds to the lowest layer (closest to the input). VGG11/13/19 in Fig. 18.

time, we don't need to worry about $\|W_r\|$ which grows with more over-parameterization. Moreover, random initialization often gives orthogonal weight vectors, which naturally leads to Assumption 3.

3.4 Extension to Multi-layer ReLU networks

Using a similar approach, we could extend this analysis to multi-layer cases. We *conjecture* that similar behaviors happen: for each layer, due to over-parameterization, the weights of some *lucky* student nodes are close to the teacher ones. While these converge to the teacher, the final values of others *irrelevant* weights are initialization-dependent. If the irrelevant nodes connect to lucky nodes at the upper-layer, then similar to Thm. 5, the corresponding fan-out weights converge to zero. On the other hand, if they connect to nodes that are also irrelevant, then these fan-out weights are not-determined and their final values depends on initialization. However, it doesn't matter since these upper-layer irrelevant nodes eventually meet with zero weights if going up recursively, since the top-most output layer has no over-parameterization. We leave a formal analysis to future work.

4 Simulations

4.1 Checking Assumption 3

To make Theorem 4 and Theorem 5 work, we make Assumption 3 that the activation field of different teacher nodes should be well-separated. To justify this, we analyze the bias of BatchNorm layers after the convolutional layers in pre-trained VGG11/13/16/19. We check the BatchNorm bias c_1 as these models use Linear-BatchNorm-ReLU architecture. After BatchNorm first normalizes the input data into zero mean distribution, the BatchNorm bias determines how much data pass the ReLU threshold. If the bias is negative, then a small portion of data pass ReLU gating and Assumption 3 is likely to hold. From Fig. 6, it is quite clear that the majority of BatchNorm bias parameters are negative, in particular for the top layers.

4.2 Numerical Experiments of Thm. 5

We verify Thm. 5 by checking whether V_r moves close to 0 under different initialization. We use a network with one hidden layer. The teacher network is 10-20-30, while the student network has more nodes in the hidden layers. Input data are Gaussian noise. We initialize the student networks so that the first 20 nodes are close to the teacher. Specifically, we first create matrices W_ϵ and V_ϵ by first filling with i.i.d Gaussian noise, and then normalizing their columns to 1. Then the initial value of student W is $W_u^0 = \text{column\_normalize}(p_W W^* + W_\epsilon)$, where p_W is a factor controlling how close W_u^0 is to W^*. For W_r we initialize with noise. Similarly we initialize V_u with a factor p_V. The larger p_W and p_V, the close the initialization W_u^0 and V_u^0 to the ground truth values.

Fig. 7 shows the behavior over different iterations. All experiments are repeated 32 times with different random seeds, and (mean$\pm$ std) are reported. We can see that a close initialization leads to faster (and low variance) convergence of V_r to small values. In particular, it is important to have W_u^0 close to W^* (large p_W), which leads to a clear separation between row norms of V_u and V_r, even if they are close to each other at the beginning of training. Having V_u^0 close to V^* makes the initial gap larger and also helps convergence. On the other hand, if p_W is small, then even if p_V is large, the gap between row norms of V_u and V_r only shifts but doesn't expand over iterations.

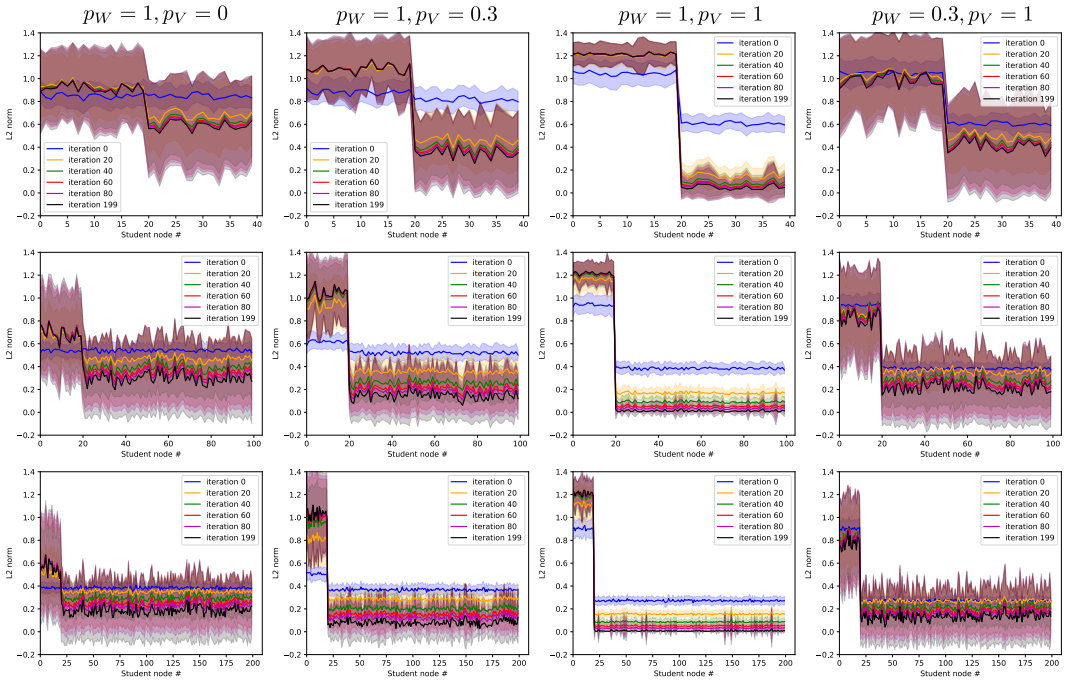


Figure 7: Numerical verification of Thm. 5. Rows are $2\times$, $5\times$ and $10\times$ over-parameterization. Columns are different initializations of student networks. All figures show $\|\mathbf{v}_j\|$, i.e., norm of each row of V.

5 Experiments

5.1 Experiment Setup

We evaluate both the fully connected (FC) and ConvNet setting. For FC, we use a ReLU teacher network of size 50-75-100-125. For ConvNet, we use a teacher with channel size 64-64-64-64. The student networks have the same depth but with $10\times$ nodes/channels at each layer, such that they are substnatially over-parameterized. When BatchNorm is added, it is added after ReLU.

We use random i.i.d Gaussian inputs with mean 0 and std 10 (abbreviated as `GAUS`) and `CIFAR-10` as our dataset in the experiments. `GAUS` generates infinite number of samples while `CIFAR-10` is a finite dataset. For `GAUS`, we use a random teacher network as the label provider (with 100 classes). To make sure the weights of the teacher are weakly overlapped, we sample each entry of $\mathbf{w}_j^*$ from $[-0.5, -0.25, 0, 0.25, 0.5]$, making sure they are non-zero and mutually different within the same layer, and sample biases from $U[-0.5, 0.5]$. In the FC case, the data dimension is 20 while in the ConvNet case it is 16×16. For `CIFAR-10` we use a pre-trained teacher network with BatchNorm. In the FC case, it has an accuracy of 54.95%; for ConvNet, the accuracy is 86.4%. We repeat 5 times for all experiments, with different random seed and report min/max values.

Two metrics are used to check our prediction that some lucky student nodes converge to the teacher:

Normalized correlation $\bar{\rho}$. We compute normalized correlation (or cosine similarity) ρ between teacher and student activations[1] evaluated on a validation set. At each layer, we average the best correlation over teacher nodes: $\bar{\rho} = \text{mean}_{j^\circ} \max_j \rho_{jj^\circ}$, where ρ_{jj° is computed for each teacher and student pairs (j, j°). $\bar{\rho} \approx 1$ means that most teacher nodes are covered by at least one student.

Mean Rank $\bar{r}$. After training, each teacher node j° has the most correlated student node j. We check the correlation rank of j, normalized to $[0, 1]$ (0=rank first), back at initialization and at different epochs, and average them over teacher nodes to yield mean rank $\bar{r}$. Small $\bar{r}$ means that student nodes that initially correlate well with the teacher keeps the lead toward the end of training.

[1]For $\mathbf{f}_j = [f_j(x_1), \ldots, f_j(x_n)]$ and $\mathbf{f}_{j^\circ}$, $\rho_{jj^\circ} = \tilde{\mathbf{f}}_j^T \tilde{\mathbf{f}}_{j^\circ} \in [-1, 1]$, where $\tilde{\mathbf{f}}_j = (\mathbf{f}_j - \text{mean}(\mathbf{f}_j))/\text{std}(\mathbf{f}_j)$.

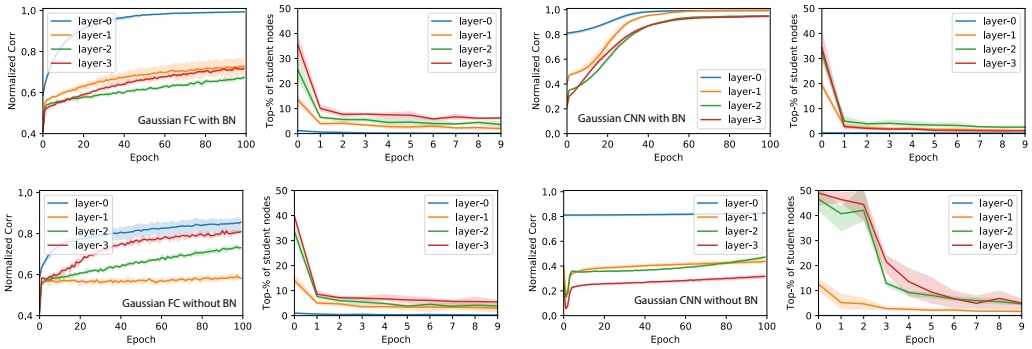


Figure 8: Correlation $\bar{\rho}$ and mean rank $\bar{r}$ over training on `GAUS`. $\bar{\rho}$ steadily grows and $\bar{r}$ quickly improves over time. `Layer-0` (the lowest layer that is closest to the input) shows best match with teacher nodes and best mean rank. BatchNorm helps achieve both better correlation and lower $\bar{r}$, in particular for the CNN case.

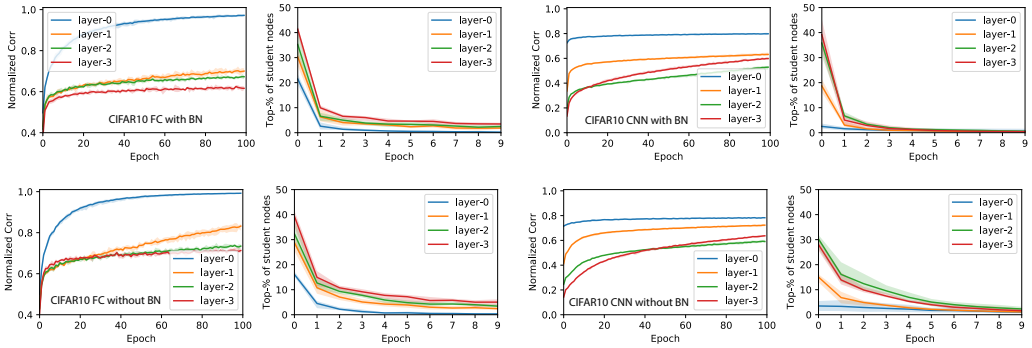


Figure 9: Same experiment setting as in Fig. 8 on `CIFAR-10`. BatchNorm helps achieve lower $\bar{r}$.

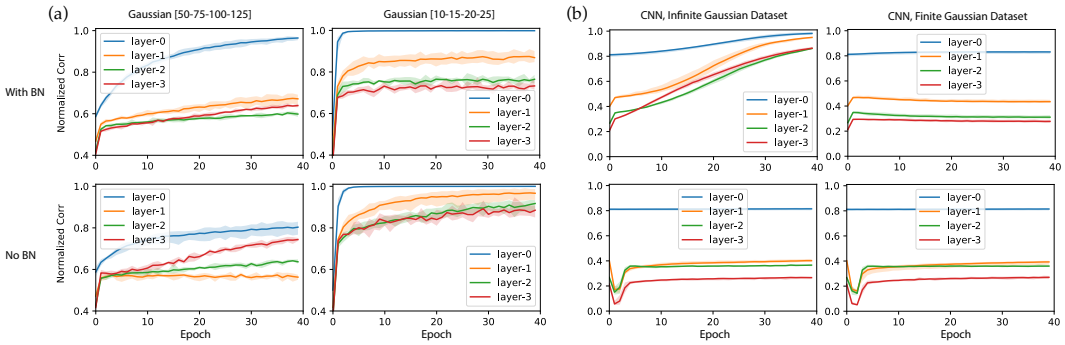


Figure 10: Ablation studies on `GAUS`. **(a)** $\bar{\rho}$ converges much faster in small models (10-15-20-25) than in large model (50-75-100-125). BatchNorm hurts in small models. **(b)** $\bar{\rho}$ stalls using finite samples.

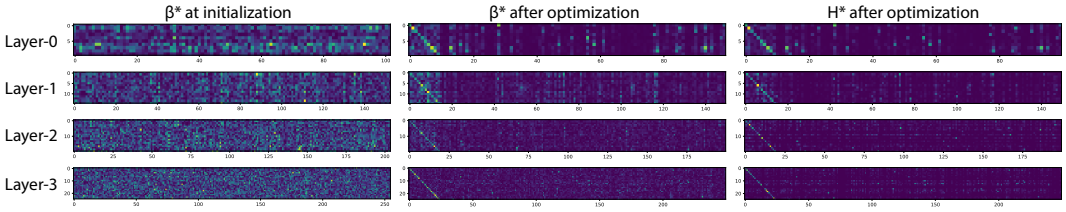


Figure 11: Visualization of (transpose of) H^* and $\boldsymbol{\beta}^*$ matrix before and after optimization (using GAUS). Student node indices are reordered according to teacher-student node correlations. After optimization, student node who has high correlation with the teacher node also has high β entries. Such a behavior is more prominent in H^* matrix that combines $\boldsymbol{\beta}^*$ and the activation patterns D^* of student nodes (Sec. 2).

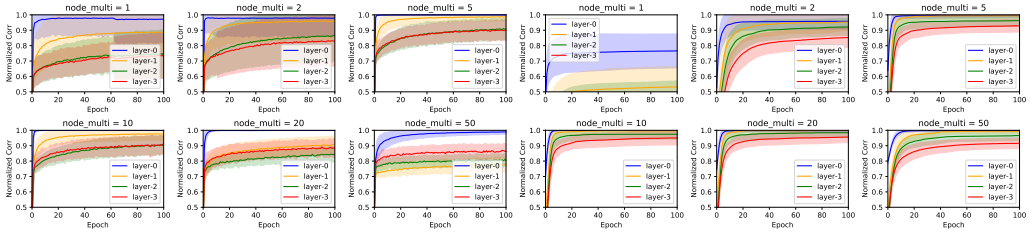


Figure 12: Effect of different degrees of over-parameterization. Left panel: Teacher FC (10-15-20-25) without batchnorm, right panel: teacher CNN (10-15-20-25) with batchnorm.

5.2 Results

Experiments are summarized in Fig. 8 and Fig. 9. $\bar{\rho}$ indeed grows during training, in particular for low layers that are closer to the input, where $\bar{\rho}$ moves towards 1. Furthermore, the final winning student nodes also have a good rank at the early stage of training, in particular after the first epoch, which is consistent with *late-resetting* used in [8]. BatchNorm helps a lot, in particular for the CNN case with GAUS dataset. For CIFAR-10, the final evaluation accuracy (see Appendix) learned by the student is often $\sim 1\%$ higher than the teacher. Using BatchNorm accelerates the growth of accuracy, improves $\bar{r}$, but seems not to accelerate the growth of $\bar{\rho}$.

The theory also predicts that the top-down modulation $\boldsymbol{\beta}$ helps the convergence. For this, we plot $\beta^*_{jj^\circ}$ at different layers during optimization on GAUS. For better visualization, we align each student node index j with a teacher node j° according to highest ρ. Despite the fact that correlations are computed from the low-layer weights, it matches well with the top-layer modulation (identity matrix structure in Fig. 11). Besides, we also perform ablation studies on GAUS.

Size of teacher network. As shown in Fig. 10(a), for small teacher networks (FC 10-15-20-25), the convergence is much faster and training without BatchNorm is faster than training with BatchNorm. For large teacher networks, BatchNorm definitely increases convergence speed and growth of $\bar{\rho}$.

Degree of over-parameterization. Fig. 12 shows the effects of different degree of over-parameterization ($1\times$, $2\times$, $5\times$, $10\times$, $20\times$ and $50\times$). We initialize 32 different teacher network (10-15-20-25) with different random seed, and plot $\pm$ standard derivation with shaded area. We can clearly see that $\bar{\rho}$ grows more stably and converges to higher values with over-parameterization. On the other hand, $20\times$ and $50\times$ are slower in convergence due to excessive parameters.

Finite versus Infinite Dataset. We also repeat the experiments with a pre-generated finite dataset of GAUS in the CNN case (Fig. 10(b)), and find that the convergence of node similarity stalls after a few iterations. This is because some nodes receive very few data points in their activated regions, which is not a problem for infinite dataset. We suspect that this is probably the reason why CIFAR-10, as a finite dataset, does not show similar behavior as GAUS.

6 Conclusion and Future Work

In this paper we propose a new theoretical framework that uses teacher-student setting to understand the training dynamics of multi-layered ReLU network. With this framework, we are able to conceptually explain many puzzling phenomena in deep networks, such as why over-parameterization helps generalization, why the same network can fit to both random and structured data, why lottery

tickets [7, 8] exist. We backup these intuitive explanations by Theorem 4 and Theorem 5, which collectively show that student nodes that are initialized to be close to the teacher nodes converge to them with a faster rate, and the fan-out weights of irrelevant nodes converge to zero. As the next steps, we aim to extend Theorem 5 to general multi-layer setting (when both L and H are present), relax Assumption 3 and study more BatchNorm effects than what Theorem 3 suggests.

7 Acknowledgement

The first author thanks Simon Du, Jason Lee, Chiyuan Zhang, Rong Ge, Greg Yang, Jonathan Frankle and many others for the informal discussions.

References

[1] Dario Amodei, Sundaram Ananthanarayanan, Rishita Anubhai, Jingliang Bai, Eric Battenberg, Carl Case, Jared Casper, Bryan Catanzaro, Qiang Cheng, Guoliang Chen, et al. Deep speech 2: End-to-end speech recognition in english and mandarin. In *International conference on machine learning*, pages 173–182, 2016.

[2] Sanjeev Arora, Zhiyuan Li, and Kaifeng Lyu. Theoretical analysis of auto rate-tuning by batch normalization. *arXiv preprint arXiv:1812.03981*, 2018.

[3] Marco Baity-Jesi, Levent Sagun, Mario Geiger, Stefano Spigler, G Ben Arous, Chiara Cammarota, Yann LeCun, Matthieu Wyart, and Giulio Biroli. Comparing dynamics: Deep neural networks versus glassy systems. *ICML*, 2018.

[4] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[5] Laurent Dinh, Razvan Pascanu, Samy Bengio, and Yoshua Bengio. Sharp minima can generalize for deep nets. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 1019–1028. JMLR. org, 2017.

[6] Simon S Du, Jason D Lee, Yuandong Tian, Barnabas Poczos, and Aarti Singh. Gradient descent learns one-hidden-layer cnn: Don't be afraid of spurious local minima. *ICML*, 2018.

[7] Jonathan Frankle and Michael Carbin. The lottery ticket hypothesis: Training pruned neural networks. *ICLR*, 2019.

[8] Jonathan Frankle, Gintare Karolina Dziugaite, Daniel M Roy, and Michael Carbin. The lottery ticket hypothesis at scale. *arXiv preprint arXiv:1903.01611*, 2019.

[9] Song Han, Jeff Pool, John Tran, and William Dally. Learning both weights and connections for efficient neural network. In *Advances in neural information processing systems*, pages 1135–1143, 2015.

[10] Babak Hassibi, David G Stork, and Gregory J Wolff. Optimal brain surgeon and general network pruning. In *IEEE international conference on neural networks*, pages 293–299. IEEE, 1993.

[11] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[12] Geoffrey Hinton, Li Deng, Dong Yu, George Dahl, Abdel-rahman Mohamed, Navdeep Jaitly, Andrew Senior, Vincent Vanhoucke, Patrick Nguyen, Brian Kingsbury, et al. Deep neural networks for acoustic modeling in speech recognition. *IEEE Signal processing magazine*, 29, 2012.

[13] Sepp Hochreiter and Jürgen Schmidhuber. Flat minima. *Neural Computation*, 9(1):1–42, 1997.

[14] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *ICML*, 2015.

[15] Nitish Shirish Keskar, Dheevatsa Mudigere, Jorge Nocedal, Mikhail Smelyanskiy, and Ping Tak Peter Tang. On large-batch training for deep learning: Generalization gap and sharp minima. *ICLR*, 2017.

[16] Jonas Kohler, Hadi Daneshmand, Aurelien Lucchi, Ming Zhou, Klaus Neymeyr, and Thomas Hofmann. Towards a theoretical understanding of batch normalization. *arXiv preprint arXiv:1805.10694*, 2018.

[17] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[18] David Krueger, Nicolas Ballas, Stanislaw Jastrzebski, Devansh Arpit, Maxinder S Kanwal, Tegan Maharaj, Emmanuel Bengio, Asja Fischer, and Aaron Courville. Deep nets don't learn via memorization. *ICLR Workshop*, 2017.

[19] Yann LeCun, John S Denker, and Sara A Solla. Optimal brain damage. In *Advances in neural information processing systems*, pages 598–605, 1990.

[20] Chunyuan Li, Heerad Farkhoor, Rosanne Liu, and Jason Yosinski. Measuring the intrinsic dimension of objective landscapes. *ICLR*, 2018.

[21] Hao Li, Zheng Xu, Gavin Taylor, Christoph Studer, and Tom Goldstein. Visualizing the loss landscape of neural nets. In *Advances in Neural Information Processing Systems*, pages 6389–6399, 2018.

[22] Zachary C Lipton. Stuck in a what? adventures in weight space. *arXiv preprint arXiv:1602.07320*, 2016.

[23] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119, 2013.

[24] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*, 2013.

[25] Behnam Neyshabur, Srinadh Bhojanapalli, David McAllester, and Nati Srebro. Exploring generalization in deep learning. In *Advances in Neural Information Processing Systems*, pages 5947–5956, 2017.

[26] Behnam Neyshabur, Zhiyuan Li, Srinadh Bhojanapalli, Yann LeCun, and Nathan Srebro. Towards understanding the role of over-parametrization in generalization of neural networks. *arXiv preprint arXiv:1805.12076*, 2018.

[27] Behnam Neyshabur, Ryota Tomioka, and Nathan Srebro. In search of the real inductive bias: On the role of implicit regularization in deep learning. *ICLR Workshop*, 2015.

[28] David Rolnick, Andreas Veit, Serge Belongie, and Nir Shavit. Deep learning is robust to massive label noise. *arXiv preprint arXiv:1705.10694*, 2017.

[29] Levent Sagun, Leon Bottou, and Yann LeCun. Eigenvalues of the hessian in deep learning: Singularity and beyond. *ICLR*, 2017.

[30] Levent Sagun, Utku Evci, V Ugur Guney, Yann Dauphin, and Leon Bottou. Empirical analysis of the hessian of over-parametrized neural networks. *ICLR 2018 Workshop Contribution, arXiv preprint arXiv:1706.04454*, 2018.

[31] David Silver, Aja Huang, Chris J Maddison, Arthur Guez, Laurent Sifre, George Van Den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, et al. Mastering the game of go with deep neural networks and tree search. *nature*, 529(7587):484, 2016.

[32] David Silver, Julian Schrittwieser, Karen Simonyan, Ioannis Antonoglou, Aja Huang, Arthur Guez, Thomas Hubert, Lucas Baker, Matthew Lai, Adrian Bolton, et al. Mastering the game of go without human knowledge. *Nature*, 550(7676):354, 2017.

[33] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[34] Stefano Spigler, Mario Geiger, Stéphane d'Ascoli, Levent Sagun, Giulio Biroli, and Matthieu Wyart. A jamming transition from under-to over-parametrization affects loss landscape and generalization. *arXiv preprint arXiv:1810.09665*, 2018.

[35] Yuandong Tian. A theoretical framework for deep locally connected relu network. *arXiv preprint arXiv:1809.10829*, 2018.

[36] Yuandong Tian and Yan Zhu. Better computer go player with neural network and long-term prediction. *ICLR*, 2016.

[37] Lei Wu, Zhanxing Zhu, et al. Towards understanding generalization of deep learning: Perspective of loss landscapes. *arXiv preprint arXiv:1706.10239*, 2017.

[38] Greg Yang, Jeffrey Pennington, Vinay Rao, Jascha Sohl-Dickstein, and Samuel S Schoenholz. A mean field theory of batch normalization. *ICLR*, 2019.

[39] Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. Understanding deep learning requires rethinking generalization. *ICLR*, 2017.

[40] Chiyuan Zhang, Samy Bengio, Moritz Hardt, and Yoram Singer. Identity crisis: Memorization and generalization under extreme overparameterization. *arXiv preprint arXiv:1902.04698*, 2019.

[41] Hattie Zhou, Janice Lan, Rosanne Liu, and Jason Yosinski. Deconstructing lottery tickets: Zeros, signs, and the supermask. *arXiv preprint arXiv:1905.01067*, 2019.

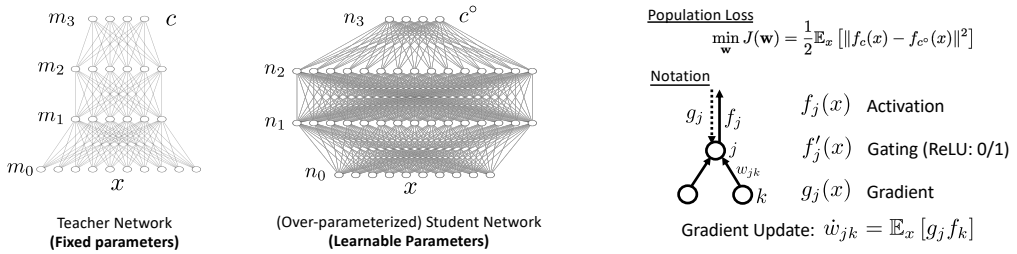


Figure 13: Teacher-Student Setting, loss function and notations.

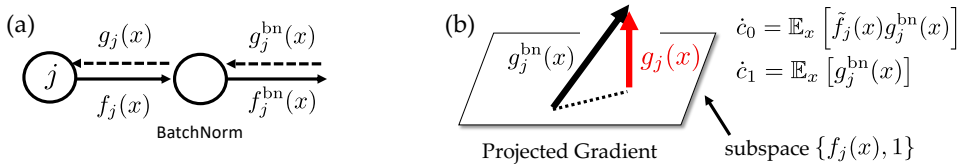


Figure 14: BatchNorm explanation

8 Appendix: Proofs

8.1 Theorem 1

Proof. The first part of gradient backpropagated to node j is:

$$g_j^1(x) = f'_j(x) \sum_{j^\circ} \beta^*_{jj^\circ}(x) f_{j^\circ}(x) \tag{10}$$

$$= f'_j(x) \sum_{j^\circ} \beta^*_{jj^\circ}(x) f'_{j^\circ}(x) \sum_{k^\circ} w^*_{j^\circ k^\circ} f_{k^\circ}(x) \tag{11}$$

$$= \sum_{k^\circ} \left[f'_j(x) \sum_{j^\circ} \beta^*_{jj^\circ}(x) f'_{j^\circ}(x) w^*_{j^\circ k^\circ} \right] f_{k^\circ}(x) \tag{12}$$

Therefore, for the gradient to node k, we have:

$$g_k^1(x) = f'_k(x) \sum_j w_{jk} g_j^1(x) \tag{13}$$

$$= f'_k(x) \sum_{k^\circ} \underbrace{\left[\sum_{jj^\circ} w_{jk} f'_j(x) \beta^*_{jj^\circ}(x) f'_{j^\circ}(x) w^*_{j^\circ k^\circ} \right]}_{\beta^*_{kk^\circ}(x)} f_{k^\circ}(x) \tag{14}$$

And similar for $\beta_{kk'}(x)$. Therefore, by mathematical induction, we know that all gradient at nodes in different layer follows the same form. □

8.2 Theorem 2

Proof. Using Thm. 1, we can write down weight update for weight w_{jk} that connects node j and node k:

$$\dot{w}_{jk} = \sum_{j^\circ, k^\circ} w^*_{j^\circ k^\circ} \underbrace{\mathbb{E}_x\left[f'_j(x) \beta^*_{jj^\circ}(x) f'_{j^\circ}(x) f_k(x) f_{k^\circ}(x) \right]}_{\beta^*_{jj^\circ kk^\circ}}$$

$$- \sum_{j', k'} w_{j'k'} \underbrace{\mathbb{E}_x\left[f'_j(x) \beta_{jj'}(x) f'_{j'}(x) f_k(x) f_{k'}(x) \right]}_{\beta_{jj'kk'}} \tag{15}$$

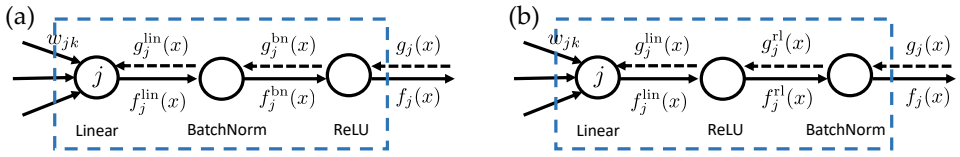


Figure 15: Different BatchNorm Configuration.

Note that j°, k°, j' and k' run over all parents and children nodes on the teacher side. This formulation works for over-parameterization (e.g., j° and j' can run over different nodes). Applying Assumption 1 and rearrange terms in matrix form yields Eqn. 6. □

8.3 Theorem 3

Proof. Given a batch with size N, denote pre-batchnorm activations as $\mathbf{f} = [f_j(x_1), \ldots, f_j(x_N)]^T$ and gradients as $\mathbf{g} = [g_j(x_1), \ldots, g_j(x_N)]^T$ (See Fig. 14(a)). $\tilde{\mathbf{f}} = (\mathbf{f} - \mu)/\sigma$ is its whitened version, and $c_0\tilde{\mathbf{f}} + c_1$ is the final output of BN. Here $\mu = \frac{1}{N}\sum_i f_j(x_i)$ and $\sigma^2 = \frac{1}{N}\sum_i (f_j(x_i) - \mu)^2$ and c_1, c_0 are learnable parameters. With vector notation, the gradient update in BN has a compact form with clear geometric meaning:

Lemma 1 (Backpropagation of Batch Norm [35])**.** *For a top-down gradient* $\mathbf{g}$*, BN layer gives the following gradient update (*$P^{\perp}_{\mathbf{f},\mathbf{1}}$ *is the orthogonal complementary projection of subspace* $\{\mathbf{f}, \mathbf{1}\}$*):*

$$\mathbf{g_f} = J^{BN}(\mathbf{f})\mathbf{g} = \frac{c_0}{\sigma} P^{\perp}_{\mathbf{f},\mathbf{1}}\mathbf{g}, \quad \mathbf{g_c} = S(\mathbf{f})^T\mathbf{g} \tag{16}$$

Intuitively, the back-propagated gradient $J^{BN}(\mathbf{f})\mathbf{g}$ is zero-mean and perpendicular to the input activation $\mathbf{f}$ of BN layer, as illustrated in Fig. 14. Unlike [16, 38] that analyzes BN in an approximate manner, in Thm. 1 we do not impose any assumptions.

Given Lemma 1, we can prove Thm. 3. For Fig. 15(a), using the property that $\mathbb{E}_x\left[g_j^{\text{lin}} f_j^{\text{lin}}\right] = 0$ (the expectation is taken over batch) and the weight update rule $\dot{w}_{jk} = \mathbb{E}_x\left[g_j^{\text{lin}} f_k\right]$ (over the same batch), we have:

$$\frac{1}{2}\frac{\mathrm{d}\|\mathbf{w}_j\|^2}{\mathrm{d}t} = \sum_{k\in\text{ch}(j)} w_{jk}\dot{w}_{jk} = \mathbb{E}_x\left[\sum_{k\in\text{ch}(j)} w_{jk} f_k(x) g_j^{\text{lin}}(x)\right] = \mathbb{E}_x\left[f_j^{\text{lin}}(x) g_j^{\text{lin}}(x)\right] = 0 \tag{17}$$

For Fig. 15(b), note that $\mathbb{E}_x\left[g_j^{\text{lin}} f_j^{\text{lin}}\right] = \mathbb{E}_x\left[g_j^{\text{rl}} f_j^{\text{rl}'} f_j^{\text{lin}}\right] = \mathbb{E}_x\left[g_j^{\text{rl}} f_j^{\text{rl}}\right] = 0$ and conclusion follows. □

8.4 Lemmas

For simplicity, in the following, we use $\delta\mathbf{w}_j = \mathbf{w}_j - \mathbf{w}_j^*$.

Lemma 2 (Bottom Bounds)**.** *Assume all* $\|\mathbf{w}_j\| = \|\mathbf{w}_{j'}\| = 1$*. Denote*

$$\mathbf{p}^*_{jj'} \equiv \mathbf{w}^*_{j'} d^*_{jj'}, \quad \mathbf{p}_{jj'} \equiv \mathbf{w}_{j'} d_{jj'}, \quad \Delta\mathbf{p}_{jj'} \equiv \mathbf{p}^*_{jj'} - \mathbf{p}_{jj'} \tag{18}$$

If Assumption 2 holds, we have:

$$\|\Delta\mathbf{p}_{jj'}\| \le (1 + K_d) d^*_{jj'} \|\delta\mathbf{w}_{j'}\| \tag{19}$$

If Assumption 3 also holds, then:

$$d^*_{jj'} \le \epsilon_d (1 + K_d\|\delta\mathbf{w}_{j'}\|)(1 + K_d\|\delta\mathbf{w}_j\|) d^*_{jj} \tag{20}$$

Proof. We have for $j \neq j'$:

$$\begin{aligned}
\|\Delta\mathbf{p}_{jj'}\| &= \|\mathbf{w}^*_{j'} d^*_{jj'} - \mathbf{w}_{j'} d_{jj'}\| && (21)\\
&= \|\mathbf{w}_{j'}(d^*_{jj'} - d_{jj'}) + (\mathbf{w}^*_{j'} - \mathbf{w}_{j'}) d^*_{jj'}\| && (22)\\
&\le \|\mathbf{w}_{j'}\| \|d^*_{jj'} - d_{jj'}\| + \|\mathbf{w}^*_{j'} - \mathbf{w}_{j'}\| d^*_{jj'} && (23)\\
&\le d^*_{jj'} K_d \|\delta\mathbf{w}_{j'}\| + d^*_{jj'} \|\delta\mathbf{w}_{j'}\| && (24)\\
&\le (1 + K_d) d^*_{jj'} \|\delta\mathbf{w}_{j'}\| && (25)
\end{aligned}$$

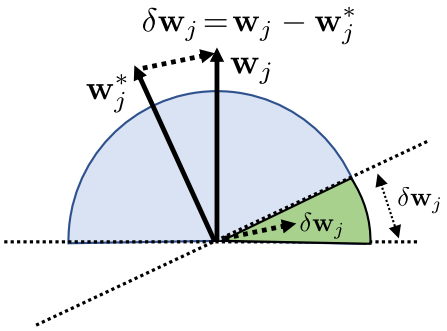


Figure 16: Explanation of Lemma. 4.

If Assumption 3 also holds, we have:

$$\begin{aligned} d^*_{jj'} &\leq d^{**}_{jj'}(1 + K_d\|\delta\mathbf{w}_{j'}\|) && (26)\\ &\leq \epsilon_d d^{**}_{jj}(1 + K_d\|\delta\mathbf{w}_{j'}\| && (27)\\ &\leq \epsilon_d d^*_{jj}(1 + K_d\|\delta\mathbf{w}_j\|)(1 + K_d\|\delta\mathbf{w}_{j'}\|) && (28)\end{aligned}$$

□

Lemma 3 (Top Bounds)**.** *Denote*

$$\mathbf{q}^*_{jj'} \equiv \mathbf{v}^*_{j'} l^*_{jj'}, \quad \mathbf{q}_{jj'} \equiv \mathbf{v}_{j'} l_{jj'}, \quad \Delta\mathbf{q}_{jj'} \equiv \mathbf{q}^*_{jj'} - \mathbf{q}_{jj'} \tag{29}$$

If Assumption 2 holds, we have:

$$\|\Delta\mathbf{q}_{jj'}\| \leq (1 + K_l) l^*_{jj'} \|\delta\mathbf{w}_{j'}\| \tag{30}$$

If Assumption 3 also holds, then:

$$l^*_{jj'} \leq \epsilon_l (1 + K_l\|\delta\mathbf{w}_{j'}\|)(1 + K_l\|\delta\mathbf{w}_j\|) l^*_{jj} \tag{31}$$

Proof. The proof is similar to Lemma 2. □

Lemma 4 (Quadratic fall-off for diagonal elements of L)**.** *For node j, we have:*

$$\|l^*_{jj} - l_{jj}\| \leq C_0 l^*_{jj} \|\delta\mathbf{w}_j\|^2 \tag{32}$$

Proof. The intuition here is that both the volume of the affected area and the weight difference are proportional to $\|\delta\mathbf{w}_j\|$. $\|l^*_{jj} - l_{jj}\|$ is their product and thus proportional to $\|\delta\mathbf{w}_j\|^2$. See Fig. 16. □

8.5 Theorem 4

Proof. First of all, note that $\|\delta\mathbf{w}_j\| = 2\sin\frac{\theta_j}{2} \leq 2\sin\frac{\theta_0}{2}$. So given θ_0, we also have a bound for $\|\delta\mathbf{w}_j\|$.

When $\boldsymbol{\beta} = \boldsymbol{\beta}^* = \mathbf{1}\mathbf{1}^T$, the matrix form can be written as the following:

$$\dot{\mathbf{w}}_j = P^\perp_{\mathbf{w}_j} \mathbf{w}^*_j h^*_{jj} + \sum_{j'\neq j} P^\perp_{\mathbf{w}_j} \left(\mathbf{w}^*_{j'} h^*_{jj'} - \mathbf{w}_{j'} h_{jj'}\right) = P^\perp_{\mathbf{w}_j} \mathbf{p}^*_{jj} + \sum_{j'\neq j} P^\perp_{\mathbf{w}_j} \Delta\mathbf{p}_{jj'} \tag{33}$$

by using $P^\perp_{\mathbf{w}_j}\mathbf{w}_j \equiv 0$ (and thus h_{jj} doesn't matter). Since $\|\mathbf{w}_j\|$ is conserved, it suffices to check whether the projected weight vector $P^\perp_{\mathbf{w}^*_j}\mathbf{w}_j$ of $\mathbf{w}_j$ onto the complementary space of the ground truth node $\mathbf{w}^*_j$, goes to zero:

$$P^\perp_{\mathbf{w}^*_j} \dot{\mathbf{w}}_j = P^\perp_{\mathbf{w}^*_j} P^\perp_{\mathbf{w}_j} \mathbf{p}^*_{jj} + \sum_{j'\neq j} P^\perp_{\mathbf{w}^*_j} P^\perp_{\mathbf{w}_j} \Delta\mathbf{p}_{jj'} \tag{34}$$

Denote $\theta_j = \angle(\mathbf{w}_j, \mathbf{w}_j^*)$ and a simple calculation gives that $\sin\theta_j = \|P^\perp_{\mathbf{w}_j^*}\mathbf{w}_j\|$. First we have:

$$P^\perp_{\mathbf{w}_j^*}P^\perp_{\mathbf{w}_j}\mathbf{w}_j^* = P^\perp_{\mathbf{w}_j^*}(I - \mathbf{w}_j\mathbf{w}_j^T)\mathbf{w}_j^* = -P^\perp_{\mathbf{w}_j^*}\mathbf{w}_j\mathbf{w}_j^T\mathbf{w}_j^* = -\cos\theta_j P^\perp_{\mathbf{w}_j^*}\mathbf{w}_j \tag{35}$$

From Lemma 2, we knows that

$$\|\Delta\mathbf{p}_{jj'}\| \le (1+K_d)d^*_{jj'}\|\delta\mathbf{w}_{j'}\| \le \epsilon_d(1+K_d)\left[1+2K_d\sin(\theta_0/2)\right]^2 d^*_{jj}\|\delta\mathbf{w}_{j'}\| \tag{36}$$

Note that here we have $\|\delta\mathbf{w}_{j'}\| = 2\sin\frac{\theta_j}{2} = \sin\theta_j/\cos\frac{\theta_j}{2} \le \sin\theta_j/\cos\frac{\theta_0}{2}$. We discuss finite step with very small learning rate $\eta > 0$:

$$\begin{aligned}
\sin\theta_j^{t+1} &= \|P^\perp_{\mathbf{w}_j^*}\mathbf{w}_j^{t+1}\| = \|P^\perp_{\mathbf{w}_j^*}\mathbf{w}_j^t + \eta P^\perp_{\mathbf{w}_j^*}\dot{\mathbf{w}}_j^t\| &(37)\\
&\le (1-\eta d^*_{jj}\cos\theta_j^t)\sin\theta_j^t + \eta\epsilon_d M_d\sum_{j'\neq j} d^*_{jj}\sin\theta_{j'}^t &(38)
\end{aligned}$$

since $\|P^\perp_{\mathbf{w}_j^*}\| = \|P^\perp_{\mathbf{w}_j}\| = 1$. Here

$$M_d = (1+K_d)\left[1+2K_d\sin(\theta_0/2)\right]^2/\cos\frac{\theta_0}{2} \tag{39}$$

is an iteration independent constant.

We set $\gamma = \cos\theta_0 - (m-1)\epsilon_d M_d$. If $\gamma > 0$, denote a constant $\bar{d} = [1+2K_d\sin(\theta_0/2)]\min_j d^{*0}_{jj}$ and from Lemma 2 we know $d^*_{jj} \ge \bar{d}$ for all j. Then given the inductive hypothesis that $\sin\theta_j^t \le (1-\eta\bar{d}\gamma)^{t-1}\sin\theta_0$, we have:

$$\sin\theta_j^{t+1} \le (1-\eta\bar{d}\gamma)^t\sin\theta_0 \tag{40}$$

Therefore, $\sin\theta_j^t \to 0$, which means that $\mathbf{w}_j \to \mathbf{w}_j^*$. □

A few remarks:

The projection operator $P^\perp_{\mathbf{w}_j}$. Note that $P^\perp_{\mathbf{w}_j}$ is important. Intuitively, without the projection, if the same proof logic worked, one could have concluded that $\mathbf{w}$ converges to any $\alpha\mathbf{w}^*$, where α is a constant scaling factor, which is obviously wrong.

Indeed, without $P^\perp_{\mathbf{w}_j}$, there would be a term $\mathbf{w}_j^*h^*_{jj} - \mathbf{w}_j h_{jj}$ on RHS. This term breaks into $\mathbf{w}_j(h^*_{jj} - h_{jj}) + (\mathbf{w}_j^* - \mathbf{w}_j)h^*_{jj}$. Although there could exist C so that $\|h^*_{jj} - h_{jj}\| \le C\|\delta\mathbf{w}_j\|$, unlike Lemma 4, C may not be small, and convergence is not guaranteed.

8.6 Theorem 5

Proof. First, only for $j \in [u]$, we have their ground truth value $\mathbf{w}_j^*$. For $j \in [r]$, we assign $\mathbf{w}_j^* = \mathbf{w}_j^0$, i.e., their initial values. As we will see, this will make things easier.

From the assumption, we know that $\sin\theta_j \le \sin\theta_0$ for $j \in [u]$. In addition, denote that $\|\delta\mathbf{v}_j^0\| \le B_{\delta v}$ for $j \in [u]$. Denote B_v as the bound for all $\|\mathbf{v}_j^*\|$.

Now suppose we can find a $\gamma > 0$ if the following set of equations are satisfied:

$$\begin{aligned}
\gamma &\ge (B_v - B_{\delta v})\cos\theta_0 - \epsilon_d(B_v + B_{\delta v})\max(B_{d,u}, B_{d,r}) > 0 &(41)\\
\gamma &\ge 1 - \epsilon_l\max(B_{l,u}, B_{l,r}) - \kappa > 0 &(42)
\end{aligned}$$

Here

$$\bar{d} = (1 - K_d C_{d,j}) \min_j d_{jj}^{*0} > 0 \tag{43}$$

$$\bar{l} = (1 - K_l C_{l,j}) \min_j l_{jj}^{*0} > 0 \tag{44}$$

$$\bar{\lambda} = \min(\bar{d}, \bar{l}) \tag{45}$$

$$\kappa = 2C_0 \sin(\theta_0/2)(1 + B_{\delta v}) \tag{46}$$

$$C_{d,u} = 2K_d \sin(\theta_0/2) \tag{47}$$

$$C_{d,r} = \epsilon_d K_d \frac{B_{d,r}(B_v + B_{\delta v})B_v}{\bar{\lambda}\gamma(2 - \eta\bar{\lambda}\gamma)} \tag{48}$$

$$M_d^{uu} = (1 + K_d)(1 + C_{d,u})^2 / \cos\frac{\theta_0}{2} \tag{49}$$

$$M_d^{ur} = (1 + K_d)(1 + C_{d,u})(1 + C_{d,r}) \tag{50}$$

$$M_d^{ru} = (1 + K_d)(1 + C_{d,u})(1 + C_{d,r}) / \cos\frac{\theta_0}{2} \tag{51}$$

$$M_d^{rr} = (1 + K_d)(1 + C_{d,r})^2 \tag{52}$$

$$B_{d,u} = (m - 1)M_d^{uu} + (n - m)M_d^{ur} \tag{53}$$

$$B_{d,r} = (m - 1)M_d^{ru} + (n - m)M_d^{rr} \tag{54}$$

and similarly we can define C_l and M_l etc. If we can find such a $\gamma > 0$ then the dynamics converges. Here all C are close to 0 and M are close to 1.

Note that if ϵ_d and ϵ_l are small, it is obvious to see there exists a feasible $\gamma > 0$ (e.g., $\gamma = 1$).

To prove it, we maintain the following induction hypothesis for iteration t :

$$d_{jj'}^{*t} \le \epsilon_d M_{d,jj'} d_{jj}^{*t}, \;\; l_{jj'}^{*t} \le \epsilon_l M_{l,jj'} l_{jj}^{*t}, \quad j' \neq j \qquad \text{(W-Separation)}$$

$$\sin\theta_j^t \le (1 - \eta\bar{d}\gamma)^{t-1} \sin\theta_0, \quad j \in [u] \qquad (W_u\text{-Contraction})$$

$$\|\delta\mathbf{v}_j^t\| \le (1 - \eta\bar{l}\gamma)^{t-1} B_{\delta v}, \;\; j \in [u], \qquad \|\mathbf{v}_j^t\| \le (1 - \eta\bar{l}\gamma)^{t-1} B_v, \;\; j \in [r] \qquad (V\text{-Contraction})$$

Besides, the following condition is involved (but it is not part of induction hypothesis):

$$\|\mathbf{w}_j^t - \mathbf{w}_j^0\| \le C_{d,r}, \quad j \in [r] \qquad (W_r\text{-Bound})$$

$$d_{jj}^{*t} \ge d_{jj}^{*0}(1 - K_d C_{d,j}) \ge \bar{d} > 0, \quad l_{jj}^{*t} \ge l_{jj}^{*0}(1 - K_l C_{l,j}) \ge \bar{l} > 0 \tag{55}$$

The proof can be decomposed in the following three lemma.

Lemma 5 (Top-layer contraction)**.** *If* (W-Separation) *holds for t, then* (V-Contraction)*) holds for iteration $t + 1$.*

Lemma 6 (Bottom-layer contraction)**.** *If* (V-Contraction) *holds for t, then* (W_u-Contraction) *holds for $t + 1$ and* (W_r-Bound) *holds for $t + 1$.*

Lemma 7 (Weight separation)**.** *If* (W-Separation) *holds for t,* (W_r-Bound) *holds for $t + 1$ and* (W_u-Contraction) *holds for $t + 1$, then* (W-Separation) *holds for $t + 1$.*

As suggested by Fig. 17, if all the three lemmas are true then the induction hypothesis are true. □

In the following, we will prove the three lemmas.

8.6.1 Lemma 5

Proof. On the top-layer, we have $\dot{V} = L^* V^* - LV$. Denote that $V = \begin{bmatrix} \mathbf{v}_1 \\ \dots \\ \mathbf{v}_n \end{bmatrix}$, where $\mathbf{v}_j$ is the j-th row of the matrix V. For each component, we can write:

$$\dot{\mathbf{v}}_j = \mathbb{I}(j \in [u])\mathbf{q}_{jj}^* - \mathbf{q}_{jj} + \sum_{j' \neq j, j' \in [u]} \Delta\mathbf{q}_{jj'} + \sum_{j' \neq j, j' \in [r]} \mathbf{q}_{jj'} \tag{56}$$

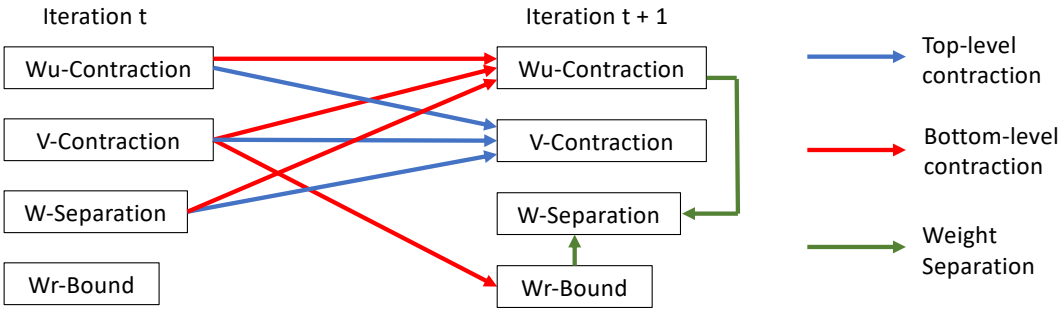


Figure 17: Proof sketch of Thm. 5.

Note that there is no projection (if there is any, the projection should be in the columns rather than the rows).

If (W-Separation) is true, we know that for $j \neq j'$,

$$\|\Delta \mathbf{q}_{jj'}\| \leq \epsilon_l M_{l,uu} l^*_{jj} \|\delta \mathbf{v}_{j'}\|, \quad \|\mathbf{q}_{jj'}\| \leq \epsilon_l M_{l,ur} l^*_{jj} \|\mathbf{v}_{j'}\|, \quad j \in [u] \tag{57}$$

$$\|\Delta \mathbf{q}_{jj'}\| \leq \epsilon_l M_{l,ru} l^*_{jj} \|\delta \mathbf{v}_{j'}\|, \quad \|\mathbf{q}_{jj'}\| \leq \epsilon_l M_{l,rr} l^*_{jj} \|\mathbf{v}_{j'}\|, \quad j \in [r] \tag{58}$$

Now we discuss $j \in [u]$ and $j \in [r]$:

Relevant nodes. For $j \in [u]$, the first two terms are:

$$\Delta \mathbf{q}_{jj} = -l^*_{jj} \delta \mathbf{v}_j + (l^*_{jj} - l_{jj}) \mathbf{v}_j \tag{59}$$

From Lemma 4 we know that:

$$\|(l^*_{jj} - l_{jj}) \mathbf{v}_j\| \leq C l^*_{jj} \|\delta \mathbf{w}_j\|^2 \|\mathbf{v}_j\| \leq 2C \sin(\theta_0/2)(1 + B_{\delta v}) l^*_{jj} \|\delta \mathbf{w}_j\| = \kappa l^*_{jj} \|\delta \mathbf{w}_j\| \tag{60}$$

Therefore using (V-Contraction) and (W_u-Contraction) at iteration t, we have:

$$\begin{aligned} \|\delta \mathbf{v}_j^{t+1}\| &\leq (1 - \eta l^*_{jj}) \|\delta \mathbf{v}_j^t\| + \eta \kappa l^*_{jj} \|\delta \mathbf{w}_j^t\| + \eta \epsilon_l M_{l,uu} l^*_{jj} \sum_{j' \neq j, j' \in [u]} \|\delta \mathbf{v}_{j'}^t\| + \eta \epsilon_l M_{l,ur} l^*_{jj} \sum_{j' \neq j, j' \in [r]} \|\mathbf{v}_{j'}^t\| \\ &\leq (1 - \eta \bar{l} \gamma)^{t+1} B_{\delta v} \end{aligned} \tag{61}$$

Since γ satisfies Eqn. 42.

Irrelevant nodes. Note that for $j \in [r]$, we don't have the term $\mathbf{q}^*_{jj}$. Therefore, we have:

$$\begin{aligned} \|\mathbf{v}_j^{t+1}\| &\leq (1 - \eta l_{jj}) \|\mathbf{v}_j^t\| + \eta \epsilon_l M_{l,ru} l^*_{jj} \sum_{j' \neq j, j' \in [u]} \|\delta \mathbf{v}_{j'}^t\| + \eta \epsilon_l M_{l,rr} l^*_{jj} \sum_{j' \neq j, j' \in [r]} \|\mathbf{v}_{j'}^t\| \\ &\leq (1 - \eta l^*_{jj}) \|\mathbf{v}_j^t\| + \eta \kappa l^*_{jj} \|\mathbf{v}_j^t\| + \eta \epsilon_l M_{l,ru} l^*_{jj} \sum_{j' \neq j, j' \in [u]} \|\delta \mathbf{v}_{j'}^t\| + \eta \epsilon_l M_{l,rr} l^*_{jj} \sum_{j' \neq j, j' \in [r]} \|\mathbf{v}_{j'}^t\| \\ &\leq (1 - \eta \bar{l} \gamma)^{t+1} B_v \end{aligned} \tag{62}$$

□

8.6.2 Lemma 6

Proof. Similar to the proof of Thm. 4, for node j, in the lower-layer, we have:

$$\dot{\mathbf{w}}_j = \mathbb{I}(j \in [u]) P^{\perp}_{\mathbf{w}_j} \tilde{\mathbf{p}}^*_{jj} + P^{\perp}_{\mathbf{w}_j} \sum_{j' \neq j, j' \in [u]} \Delta \tilde{\mathbf{p}}_{jj'} + P^{\perp}_{\mathbf{w}_j} \sum_{j' \in [r], j' \neq j} \tilde{\mathbf{p}}_{jj'} \tag{63}$$

where $h_{jj'} = d_{jj'} \mathbf{v}_j^T \mathbf{v}_{j'}$ and $\tilde{\mathbf{p}}_{jj'} = \mathbf{p}_{jj'} \mathbf{v}_j^T \mathbf{v}_{j'} = \mathbf{w}_{j'} h_{jj'}$.

Due to (W-Separation) and $\|\mathbf{w}_{j'}\| = 1$, we know that for $j \neq j'$:

$$\|\Delta \tilde{\mathbf{p}}_{jj'}\| \leq \epsilon_d M_{d,uu} d^*_{jj} \|\delta \mathbf{w}_{j'}\| \|\mathbf{v}_j\| \|\mathbf{v}_{j'}\|, \quad \|\tilde{\mathbf{p}}_{jj'}\| \leq \epsilon_d M_{d,ur} d^*_{jj} \|\delta \mathbf{w}_{j'}\| \|\mathbf{v}_j\| \|\mathbf{v}_{j'}\|, \quad j \in [u] \tag{64}$$

$$\|\Delta \tilde{\mathbf{p}}_{jj'}\| \leq \epsilon_d M_{d,ru} d^*_{jj} \|\delta \mathbf{w}_{j'}\| \|\mathbf{v}_j\| \|\mathbf{v}_{j'}\|, \quad \|\tilde{\mathbf{p}}_{jj'}\| \leq \epsilon_d M_{d,rr} d^*_{jj} \|\delta \mathbf{w}_{j'}\| \|\mathbf{v}_j\| \|\mathbf{v}_{j'}\|, \quad j \in [r] \tag{65}$$

Note that if $\|\mathbf{v}_{j'}\|$(for $j \in [r]$) doesn't converge to zero, then due to Eqn. 65, there is always residue and $\mathbf{w}_j$ won't converge to $\mathbf{w}_j^*$.

Now we discuss two cases:

Relevant nodes. For $j \in [u]$, similar to Eqn. 35 we have:

$$\begin{aligned}\sin\theta_j^{t+1} &\leq (1-\eta d_{jj}^*\|\mathbf{v}_j^t\|^2\cos\theta_j^t)\sin\theta_j^t + \eta\|\mathbf{v}_j^t\|\epsilon_d M_{d,uu} d_{jj}^* \sum_{j'\neq j, j\in[u]} \|\mathbf{v}_{j'}^t\|\sin\theta_{j'}^t \\ &+ \eta\|\mathbf{v}_j^t\|\epsilon_d M_{d,ur} d_{jj}^* \sum_{j'\neq j, j\in[r]} \|\mathbf{v}_{j'}^t\| \end{aligned} \tag{66}$$

Since (W_u-Contraction) and (V-Contraction) holds for time t, we know that:

$$\sin\theta_j^{t+1} \leq (1-\eta\bar{d}\gamma)^{t+1}\sin\theta_0 \tag{67}$$

since Eqn. 41 holds.

Irrelevant nodes. In this case, we cannot prove for $j \in [r]$, $\mathbf{w}_j$ converges to any determined target. Instead, we show that $\mathbf{w}_j$ won't move too much from its initial location $\mathbf{w}_j^0$, which is also set to be $\mathbf{w}_j^*$, before its corresponding $\mathbf{v}_j$ converges to zero. This is important to ensure that (W-Separation) remains correct thorough-out the iterations.

For any $j \in [u]$, using (W_u-Contraction) and (V-Contraction), we know that the distance between the current $\mathbf{w}_j$ and its initial value is

$$\|\mathbf{w}_j^{t+1} - \mathbf{w}_j^0\| \leq \eta\sum_{t'=0}^{t}\|\dot{\mathbf{w}}_j^{t'}\| \leq \eta\sum_{t'=0}^{t}\left\|\sum_{j'\neq j, j'\in[u]}\Delta\tilde{\mathbf{p}}_{jj'}^{t'} + \sum_{j'\in[r], j'\neq j}\tilde{\mathbf{p}}_{jj'}^{t'}\right\| \tag{68}$$

$$\leq \eta\epsilon_d B_{d,u}(B_v + B_{\delta v})B_v\sum_{t'=0}^{t}(1-\eta\bar{\lambda}\gamma)^{2t'} \tag{69}$$

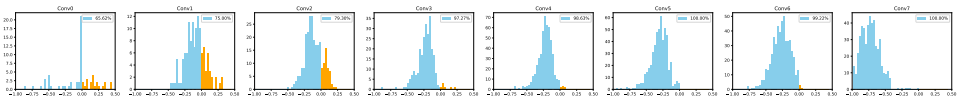
$$= \frac{\epsilon_d B_{d,r}(B_v+B_{\delta v})B_v}{\bar{\lambda}\gamma(2-\eta\bar{\lambda}\gamma)} = C_{d,r} \tag{70}$$

Therefore, we prove that (W_r-Bound) holds for iteration $t+1$. □

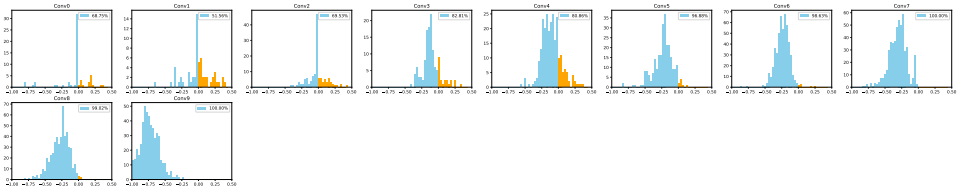
8.7 Lemma 7

Proof. Simply followed from combining Lemma 3, Lemma 2 and weight bounds (W_u-Contraction) and (V-Contraction). □

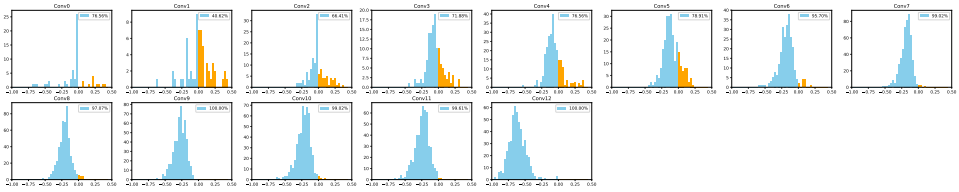
9 More experiments



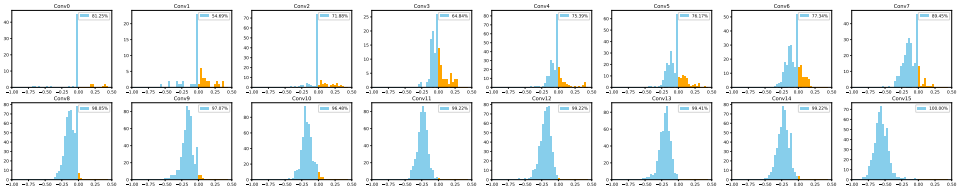
(a) VGG11



(b) VGG13



(c) VGG16



(d) VGG19

Figure 18: BatchNorm bias distribution of pre-trained VGG11/13/16/19 on ImageNet. Orange/blue are positive/negative biases. The first plot corresponds to the lowest layer (closest to the input).

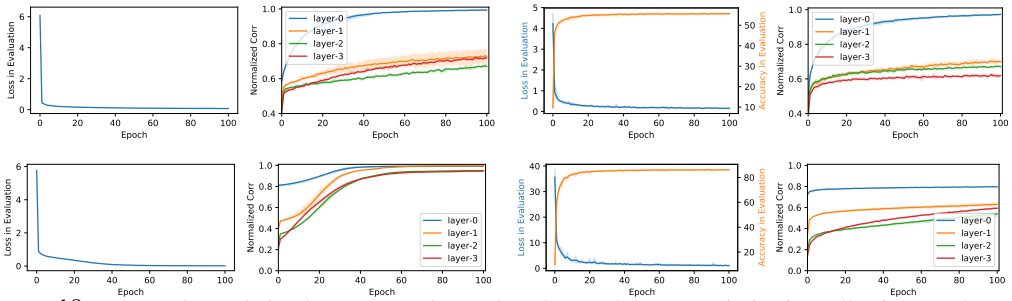


Figure 19: Loss and correlation between teacher and student nodes over optimization, all using BatchNorm. Gaussian (left) versus CIFAR10 (right). FC (top) versus CNN (bottom). `Layer-0` is the lowest layer (closest to the input). The mean best correlation steadily goes up over time.

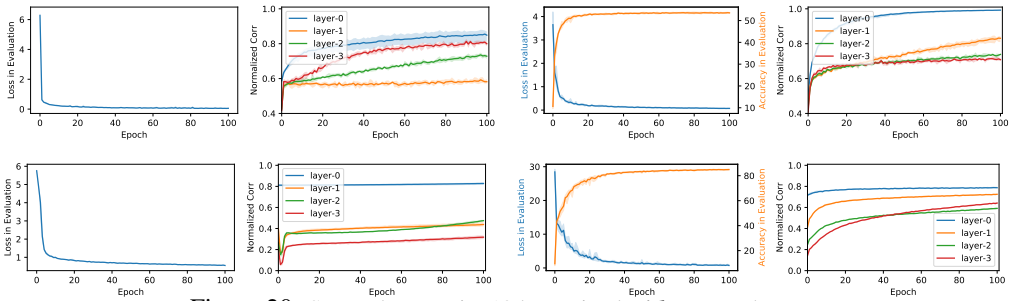


Figure 20: Same plots as Fig. 19 but trained ***without*** BatchNorm.

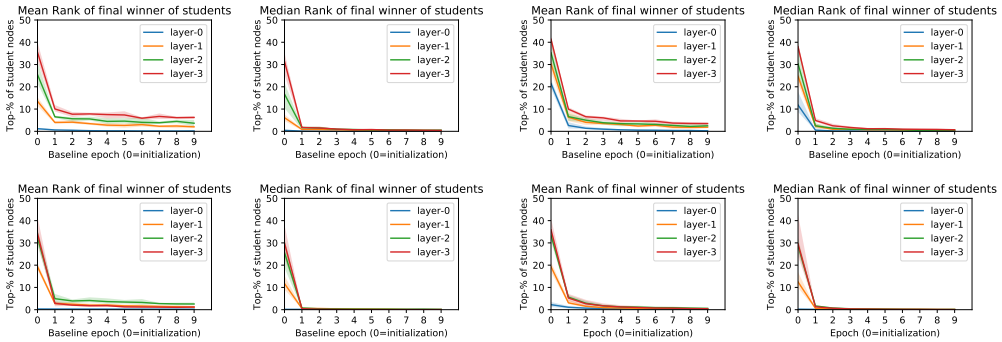


Figure 21: Mean/Median rank at different epoch of the final winning student nodes that best match the teacher nodes after the training *with* BatchNorm. Gaussian (left) and CIFAR10 (right). FC (top) and CNN (bottom). For training without BatchNorm, see Fig. 23.

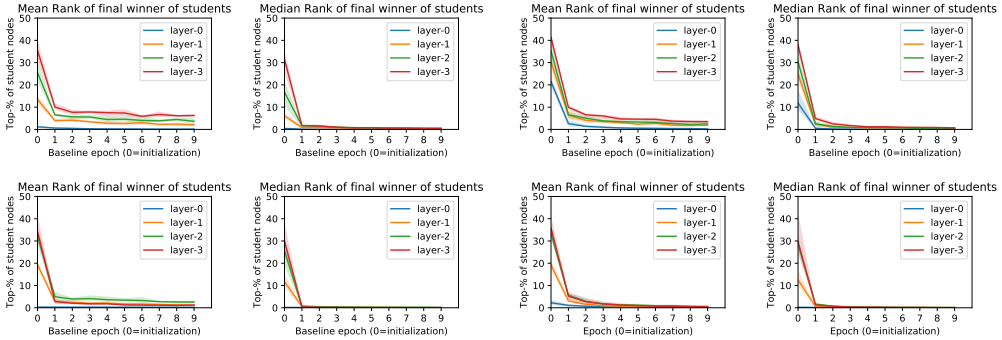


Figure 22: Mean/Median rank at different epoch of the final winning student nodes that best match the teacher nodes after the training using BatchNorm. Gaussian (left) versus CIFAR10 (right). FC (top) versus CNN (bottom).

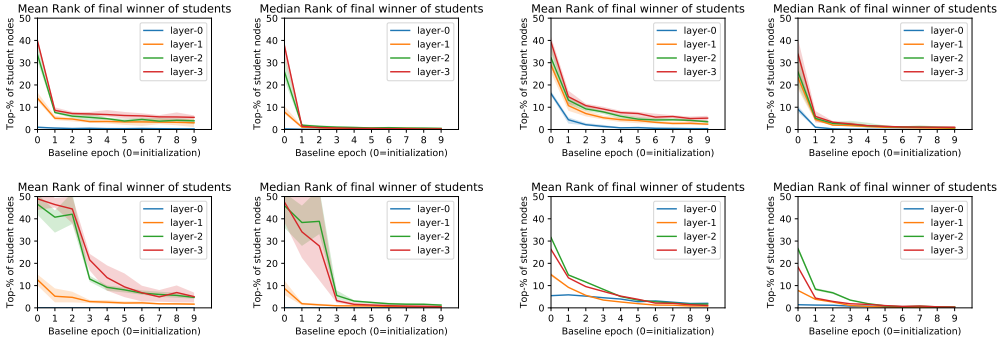


Figure 23: Mean/Median rank at different epoch of the final winning student nodes that best match the teacher nodes after the training *without* BatchNorm. Gaussian (left) versus CIFAR10 (right). FC (top) versus CNN (bottom).

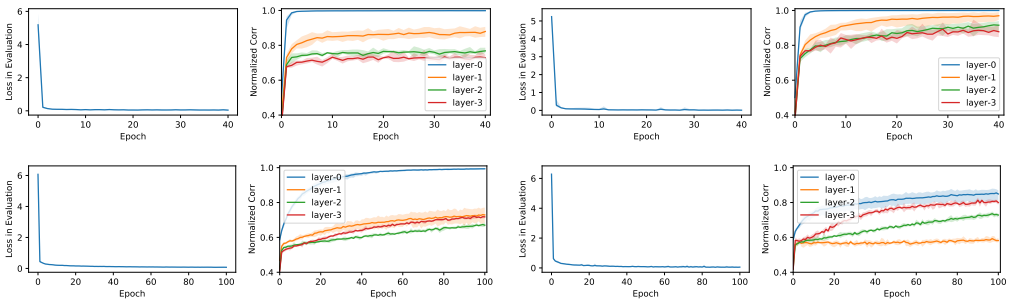


Figure 24: Gaussian data with small (10-15-20-25) and large (50-75-100-125) FC models. Small models (top) versus large models (bottom). With BN (left) versus Without BN (right).

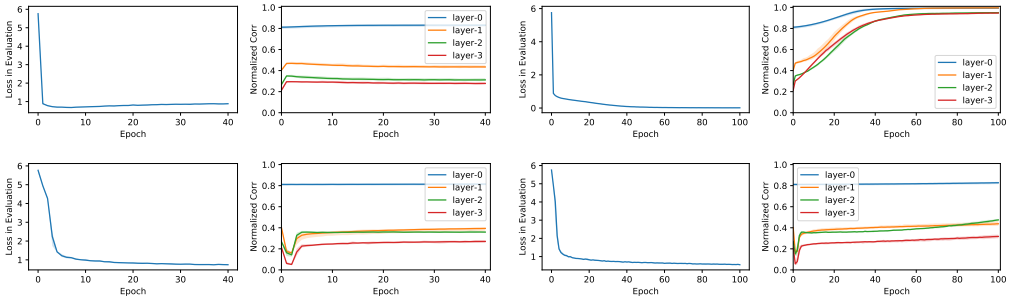


Figure 25: Gaussian CNN. With BN (top) versus Without BN (bottom). Finite Dataset (left) versus Infinite Dataset (right).