Pruning untrained neural networks : Principles and Analysis

Soufiane Hayou, Jean-Francois Ton, Arnaud Doucet, Yee Whye Teh
Department of Statistics, University of Oxford

{soufiane.hayou, ton, doucet, y.w.teh}@stats.ox.ac.uk

February 21, 2020

Abstract

Overparameterized neural networks display state-of-the art performance. However, there is a growing need for smaller, energy-efficient, neural networks to be able to use machine learning applications on devices with limited computational resources. A popular approach consists of using pruning techniques. While these techniques have traditionally focused on pruning pre-trained neural networks LeCun et al. [1990], Hassibi et al. [1993], recent work by Lee et al. [2018a] showed promising results when pruning is performed at initialization. However, such procedures remain unsatisfactory as the resulting pruned networks can be difficult to train and, for instance, they do not prevent one layer being fully pruned. In this paper we provide a comprehensive theoretical analysis of pruning at initialization and training sparse architectures. This analysis allows us to propose novel principled approaches which we validate experimentally on a variety of network architectures. In particular, we show that we can prune up to 99.9% of the weights while keeping the model trainable.

1 Introduction

Overparameterized deep neural networks have achieved state of the art performance in many tasks ranging from computer vision to speech translation Nguyen and Hein [2018], Du et al. [2019], Zhang et al. [2016], Neyshabur et al. [2019]. However, training and deploying these models in practice requires large computational power. This is problematic for a large class of embedded systems, making those methods particularly difficult to implement on small devices such as phones and tablets. To solve this

problem, network pruning has been widely used to reduce the time and space requirements both at training and test time. The general idea is to identify weights that do not contribute significantly to the model performance based on some criterion, and remove them from the architecture. However, most pruning procedures are applied to pre-trained networks and thus require training the full network LeCun et al. [1990], Hassibi et al. [1993], Mozer and Smolensky [1989], Dong et al. [2017]; this limits the benefits of these methods at training time. Another line of research has considered pruning the model during training. For example, Louizos et al. [2018] have proposed an algorithm which adds a L_0 regularization on the weights to enforce sparsity of the network while Carreira-Perpiñán and Idelbayev [2018] have developed a generic algorithm to incorporate pruning into the training task by alternating between learning and compression steps. However, these methods do not save many resources as they require the whole network during training time.

Recently, Frankle and Carbin [2019] have introduced and validated experimentally the Lottery Ticket Hypothesis which conjectures the existence of a sparse subnetwork that achieves similar performance to the original non-pruned network. These empirical findings have further motivated the search for pruning methods at initialization. Lee et al. [2018a] and Wang et al. [2020] have demonstrated that pruning at initialization could be as good as classical pruning methods which prune during or after training. Importantly, pruning at initialization does not require first training the full network and thus saves memory by only training sparse models, thus making the training of deep neural networks feasible with limited computational resources. Although the proposed pruning at initialization techniques are promising, they suffer from significant problems. In particular, nothing prevents such methods from pruning one whole layer of the network, which would thus cut off the information flow from the inputs to outputs. More generally, even in scenarios where this is not the case, it is typically difficult to train the resulting pruned networks Li et al. [2018].

In parallel, several works Hayou et al. [2019], Schoenholz et al. [2017], Poole et al. [2016], Yang and Schoenholz [2017], Xiao et al. [2018], Lee et al. [2018b], Matthews et al. [2018] have analyzed the theoretical properties of wide deep neural networks using an Mean-Field approximation by considering an infinite width limit and infinite number of channels for convolutional neural networks. This simplifies the analysis of signal propagation within the network. In Schoenholz et al. [2017] and Hayou et al. [2019], it has been shown that only one initialization, known as the Edge of Chaos, makes models trainable as the network depth goes to infinity. This theory analyses the forward and backward signal propagation to derive principled guidelines for the choice of initialization hyper-parameters. In this paper, we also rely on the Mean-Field approximation of deep neural networks to analyze gradient based pruning

at initialization. Our contribution is four-fold:

1. We define the critical sparsity s_{cr} as the maximal sparsity we can achieve without having at least one layer fully pruned. We give upper bounds on the expected value of s_{cr} for different pruning methods and show that the Edge of Chaos initialization is necessary for gradient-based pruning.

2. We show that pruning 'destroys' the Edge of Chaos, and we introduce a simple rescaling trick to bring the pruned model back into this regime..

3. We show that, unlike FeedForward neural networks (i.e. no residual connections), Residual networks are better suited for pruning at initialization since they 'live' on the Edge of Chaos by default. However, Resnets might suffer from exploding gradients Yang and Schoenholz [2017], which we resolve by introducing a re-parameterization of Resnets, called 'Stable Resnet'. It allows pruning 99.5% of ResNet104 on Cifar10 while achieving $> 87\%$ test accuracy. We can also prune up to 99.9% of the weights while keeping the model trainable, achieving 72.70% test accuracy.

4. We confirm the predictions of the Lottery Ticket Hypothesis Frankle and Carbin [2019] by showing that, starting from a wide range of randomly initialized networks, we can always find a subnetwork that is already initialized on the Edge of Chaos, and thus is trainable.

2 Neural Network Pruning

2.1 Setup and notations

Let x be an input in $\mathbb{R}^d$. In its general form, a neural network of depth L is given by the following set of forward propagation equations

$$y^l(x) = \mathcal{F}_l(W^l, y^{l-1}(x)) + B^l, \quad 1 \leq l \leq L \tag{1}$$

where $y^l(x)$ is the vector of pre-activations and W^l and B^l are respectively the weights and bias of the l^{th} layer. F_l is a mapping that defines the nature of the layer. The weights and bias are initialized with $W^l \overset{iid}{\sim} \mathcal{N}(0, \frac{\sigma_w^2}{v_l})$ where v_l is a scaling factor used to control the variance of y^l, and $B^l \overset{iid}{\sim} \mathcal{N}(0, \sigma_b^2)$. Hereafter, we denote by M_l the number of weights in the l^{th} layer, ϕ the activation function and $[|n, m|]$ the set of integers $\{n, n+1, ..., m\}$ for $n \leq m$. Two examples of such architectures are:

- **Fully-connected FeedForward Neural Network (FFNN)**
  For a fully connected feedforward neural network of depth L and widths $(N_l)_{0\leq l\leq L}$, the forward propagation of the input through the network is given by

$$\begin{aligned} y_i^1(x) &= \sum_{j=1}^{d} W_{ij}^1 x_j + B_i^1, \\ y_i^l(x) &= \sum_{j=1}^{N_{l-1}} W_{ij}^l \phi(y_j^{l-1}(x)) + B_i^l, \quad \text{for } l \geq 2. \end{aligned} \tag{2}$$

  Here, we have $v_l = N_{l-1}$ and $M_l = N_{l-1}N_l$.

- **Convolutional Neural Network (CNN/ConvNet)**
  For a 1D convolutional neural network of depth L, number of channels $(n_l)_{l\leq L}$ and number of neurons per channel $(N_l)_{l\leq L}$. we have

$$\begin{aligned} y_{i,\alpha}^1(x) &= \sum_{j=1}^{n_{l-1}} \sum_{\beta\in ker_l} W_{i,j,\beta}^1 x_{j,\alpha+\beta} + b_i^1, \\ y_{i,\alpha}^l(x) &= \sum_{j=1}^{n_{l-1}} \sum_{\beta\in ker_l} W_{i,j,\beta}^l \phi(y_{j,\alpha+\beta}^{l-1}(x)) + b_i^l, \ \text{for } l \geq 2, \end{aligned} \tag{3}$$

  where $i \in [|1, n_l|]$ is the channel index, $\alpha \in [|0, N_l - 1|]$ is the neuron location, $ker_l = [|-k_l, k_l|]$ is the filter range and $2k_l + 1$ is the filter size. To simplify the analysis, we assume hereafter that $N_l = N$ and $k_l = k$ for all l. Here, we have $v_l = n_{l-1}(2k + 1)$ and $M_l = n_{l-1}n_l(2k + 1)$. We assume periodic boundary conditions, so $y_{i,\alpha}^l = y_{i,\alpha+N}^l = y_{i,\alpha-N}^l$. Generalization to multidimensional convolutions is straighforward.

When no specific architecture is mentioned, we denote by $(W_i^l)_{1\leq i\leq M_l}$ the weights of the l^{th} layer.

Pruning overparameterized neural networks was motivated by the idea that, in this context, most of the weights do not help to significantly reduce the empirical loss. Thus, their removal would not have a large impact on model performance. In practice, a pruning algorithm creates a binary mask δ over the weights to force the pruned weights to be zero. The neural network after pruning is given by

$$y^l(x) = \mathcal{F}_l(\delta^l \circ W^l, y^{l-1}(x)) + B^l, \tag{4}$$

where $\circ$ is the Hadamard (i.e. element-wise) product. Generally, there are three approaches to creating the mask δ.

- Pruning after training: this requires training the model before pruning; e.g. LeCun et al. [1990], Hassibi et al. [1993]
- Pruning during training: pruning is alternated with training steps Louizos et al. [2018], Carreira-Perpiñán and Idelbayev [2018]
- Pruning at initialization: the network is pruned before training Lee et al. [2018a], Wang et al. [2020]

In this paper, we focus on pruning at initialization. The mask is typically created using a criterion g. More precisely, we create a vector g^l of the same dimension as W^l using a mapping of choice (see below), we then prune the network by keeping the weights that correspond to the top k values in the sequence $(g_i^l)_{i,l}$ where k is fixed by the sparsity that we want to achieve. There are generally three types of criteria.

- Magnitude based pruning (Zero-shot pruning): We prune weights that have magnitude $|W|$ less than a threshold t (t being fixed by the required sparsity). This algorithm is data independent.
- Sensitivity based pruning (One-shot pruning): We prune the weights based on the values of $|W\frac{\partial \mathcal{L}}{\partial W}|$ where $\mathcal{L}$ is the loss. This is inspired from the fact that

$$\mathcal{L}_W \approx \mathcal{L}_{W=0} + W\frac{\partial \mathcal{L}}{\partial W}.$$

  Lee et al. [2018a] used this criterion to achieve similar performance to that of non-pruned models.
- Hessian based pruning Wang et al. [2020]: We prune the weights based on the Hessian of the loss function, which is used to select weights that preserve the gradient flow.

In this work, we focus on magnitude and sensitivity based pruning and leave Hessian based methods for future work. However, we include empirical results with a Hessian based pruning method Wang et al. [2020] in Section 6.

Hereafter, we denote by s the sparsity, i.e. the fraction of weights that we have to prune, which we always assume fixed before pruning. The sparsity s has an upper bound $s_{max} = 1 - L/\sum_l M_l$, where M_l is the number of weights in the l^{th} layer. A sparsity $s > s_{\max}$ will surely result in one layer at least being fully pruned, which makes the model non trainable. Hereafter, we always assume $s < s_{\max}$, even when s is said to be in $(0, 1)$.

Let A_l be the set of indices of the weights in the l^{th} layer that are pruned, i.e. $A_l = \{i \in [|1, M_l|],\ s.t.\ \delta_i^l = 0\}$. We define the critical sparsity s_{cr} by

$$s_{cr} = \min\{s \in (0,1),\ s.t.\ \exists l, |A_l| = M_l\},$$

where $|A_l|$ is the number of elements in A_l. The critical sparsity is the maximal sparsity we can achieve without fully pruning at least one layer, in which case the model becomes non trainable. Unlike $s_{\max}$, s_{cr} is random as the weights are initialized randomly. Thus, we study the behaviour of the expected value $\mathbb{E}[s_{cr}]$ instead. This provides theoretical guidelines for pruning at initialization. Hereafter, all expectations are taken w.r.t to initialization weights.

For all $l \in [|1, L|]$, we define α_l by $v_l = \alpha_l N$ where $N > 0$ and $\zeta_l > 0$ such that $M_l = \zeta_l N^2$. Recall that v_l is a scaling factor controlling the variance of y^l and M_l is the number of weights in the l^{th} layer.

2.2 Magnitude based pruning (MBP)

Magnitude based pruning is a data independent pruning algorithm (zero-shot pruning). The mask is given by

$$\delta_i^l = \begin{cases} 1 & \text{if} \quad |W_i^l| \geq t_s, \\ 0 & \text{if} \quad |W_i^l| < t_s, \end{cases}$$

where t_s is a threshold that depends on the sparsity s. By defining $k_s = (1-s)\sum_l M_l$, t_s is given by $t_s = |W|^{(k_s)}$ where $|W|^{(k_s)}$ is the k_s^{th} order statistic of the network weights $(|W_i^l|)_{1\leq l\leq L, 1\leq i\leq M_l}$ $(|W|^{(1)} > |W|^{(2)} > ...)$.

With magnitude based pruning, changing σ_w does not impact the distribution of the resulting sparse architecture since it is a common factor for all the weights. However, in the case of different scaling factors v_l, the variances $\frac{\sigma_w^2}{v_l}$ used to initialize the weights vary across layers. This gives the false intuition that the layer with the smallest variance will be highly likely fully pruned before others as we increase the sparsity s. This is wrong in general since layers with small variances might have more weights compared to other layers. However, we can prove a similar result by considering the limit of large depth with fixed widths.

Proposition 1 (MBP in the large depth limit)**.** *Assume N is fixed and there exists $l_0 \in [|1, L|]$ such that $\alpha_{l_0} > \alpha_l$ for all $l \neq l_0$. Let Q_x be the x^{th} quantile of $|X|$ where*

$X \overset{iid}{\sim} \mathcal{N}(0,1)$ and $\gamma = \min_{l \neq l_0} \frac{\alpha_{l_0}}{\alpha_l}$. For $\epsilon \in (0,2)$, define $x_{\epsilon,\gamma} = \inf\{y \in (0,1) : \forall x > y, \gamma Q_x > Q_{1-(1-x)^{\gamma^{2-\epsilon}}}\}$ and $x_{\epsilon,\gamma} = \infty$ for the null set. Then, for all $\epsilon \in (0,2)$, $x_{\epsilon,\gamma}$ is finite and there exists a constant $\nu > 0$ such that

$$\mathbb{E}[s_{cr}] \leq \inf_{\epsilon \in (0,2)} \{x_{\epsilon,\gamma} + \frac{\zeta_{l_0} N^2}{1+\gamma^{2-\epsilon}} (1 - x_{\epsilon,\gamma})^{1+\gamma^{2-\epsilon}}\} + \mathcal{O}(\frac{1}{\sqrt{LN^2}})$$

Proposition 6 gives an upper bound on $\mathbb{E}[s_{cr}]$ in the large depth limit. The upper bound is easy to approximate numerically and our experiments reveal that it can be tight. Table 1 compares the theoretical upper bound in Proposition 6 to the empirical value of $\mathbb{E}[s_{cr}]$ over 10 simulations for a FFNN with depth $L = 100$, $N = 100$, $\alpha_1 = \gamma$ and $\alpha_2 = \alpha_3 = \cdots = \alpha_L = 1$.

Table 1: Theoretical upper bound of Proposition 6 and empirical observations for a FFNN with $N = 100$ and $L = 100$

Gamma	$\gamma = 2$	$\gamma = 5$	$\gamma = 10$
Upper bound	5.77	0.81	0.72
Empirical observation	≈ 1	0.79	0.69

In practice, MBP is not the algorithm of choice since it does not use information available from the data. Indeed, alternative, data dependent, pruning criterias such as Sensitivity and Hessian based pruning were found to be more effective in practice. Hence, in the remainder of the paper, we are going to focus and analyse Sensitivity based pruning.

2.3 Sensitivity based pruning (SBP)

Unlike magnitude based pruning, sensitivity based pruning is data-dependent. It uses the data to compute the gradient *with* backpropagation; for this reason, it is called one-shot pruning. In order to compute these gradients we randomly sample a batch and compute the gradients of the loss with respect to each weight. The mask is then defined by:

$$\delta_i^l = \begin{cases} 1 & \text{if} \quad |W_i^l \frac{\partial \mathcal{L}}{\partial W_i^l}| \geq t_s, \\ 0 & \text{if} \quad |W_i^l \frac{\partial \mathcal{L}}{\partial W_i^l}| < t_s, \end{cases}$$

where $t_s = |W \frac{\partial \mathcal{L}}{\partial W}|^{(k_s)}$ and $k_s = (1-s)\sum_l M_l$ and $|W \frac{\partial \mathcal{L}}{\partial W}|^{(k_s)}$ is the k_s^{th} order statistics of the sequence $(|W_i^l \frac{\partial \mathcal{L}}{\partial W_i^l}|)_{1 \leq l \leq L, 1 \leq i \leq M_l}$.

SBP relies on gradients at initialization to determine which weights are redundant. Neural networks might suffer from exploding or vanishing gradients which would make one layer more 'prunable' than others due to a purely structural problem. We give a formal definition to this problem.

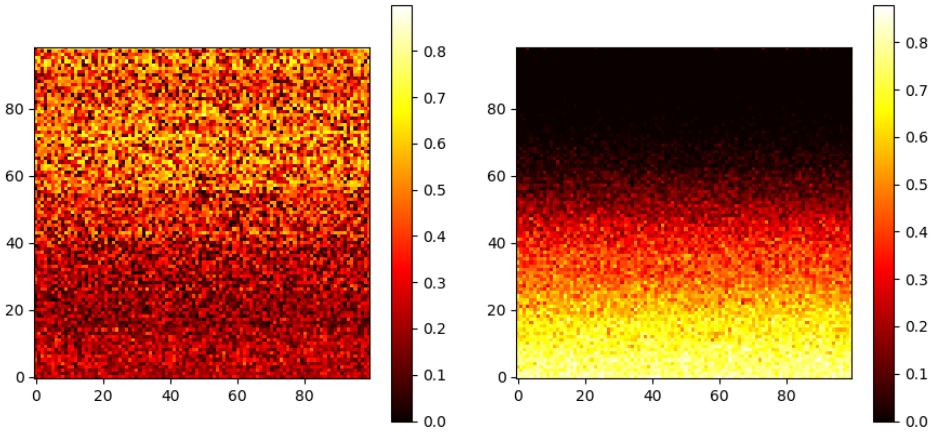
Definition 1 (Well-conditioned and ill-conditioned networks)**.** *Let* $m_l = \mathbb{E}[|W_1^l \frac{\partial \mathcal{L}}{\partial W_1^l}|^2]$ *for* $l \geq 1$*. We say that the network is well-conditioned if there exists* $A, B > 0$ *such that for all* $L \geq 1$ *and* $l \in \{1, ..., L\}$ *we have* $A \leq \frac{m_l}{m_L} \leq B$*. We say that the network is ill-conditioned otherwise.*

Understanding the behaviour of gradients at initialization is crucial for the analysis of SBP. Schoenholz et al. [2017], Hayou et al. [2019] and Xiao et al. [2018] have studied signal propagation through the network using the Mean-Field approximation (infinite width approximation), which facilitates the theoretical analysis and provides closed-form formulas. In particular, the authors showed that an initialization known as the Edge of Chaos can be beneficial for deep neural networks training as it maximizes information flow through the network.

Edge of Chaos (EOC) : For two inputs x, x', we denote by $q^l(x)$ the variance of $y^l(x)$ and $c^l(x, x')$ the correlation between $y^l(x)$ and $y^l(x')$. The asymptotic behaviour of these quantities w.r.t l is studied in Schoenholz et al. [2017] and Hayou et al. [2019]. Under weak regularity conditions, it is proved that $q^l(x)$ converges to a point $q(\sigma_b, \sigma_w) > 0$ independent of x. The asymptotic behaviour of $c^l(x, x')$ is dependent on (σ_b, σ_w). The EOC is defined as the set of parameters (σ_b, σ_w) such that $\sigma_w^2 \mathbb{E}[\phi'(\sqrt{q(\sigma_b, \sigma_w)}Z)^2] = 1$ where $Z \sim \mathcal{N}(0, 1)$. Similarly the Ordered, resp. Chaotic, phase is defined by $\sigma_w^2 \mathbb{E}[\phi'(\sqrt{q(\sigma_b, \sigma_w)}Z)^2] < 1$, resp. $\sigma_w^2 \mathbb{E}[\phi'(\sqrt{q(\sigma_b, \sigma_w)}Z)^2] > 1$. On the Ordered phase, the gradient will vanish as it backpropagates through the network, and $c^l(x, x')$ converges exponentially to 1 causing the output function to be constant (hence the name 'Ordered phase'). On the Chaotic phase, the gradient explodes and $c^l(x, x')$ converges exponentially to some limiting value $c < 1$ which results in the output function being discontinuous everywhere (hence the 'chaotic' phase name). On the EOC, the second moment of the gradient remains approximately constant throughout backpropagation and $c^l(x, x')$ converges to 1 at a sub-exponential rate, which allows deeper information propagation. The EOC can usually be represented as a curve in the 2D plan (σ_w, σ_b) that separates the Ordered phase and the Chaotic phase.

Using those results, we show in the following theorem that the initialization has a crucial impact on the pruned network with SBP.

Theorem 1 (Initialization is crucial for SBP)**.** *Consider a neural network of type (10) or (11) (FFNN or CNN). Assume* (σ_w, σ_b) *are chosen to be either on the Ordered*



(a) Edge of Chaos (b) Chaotic phase

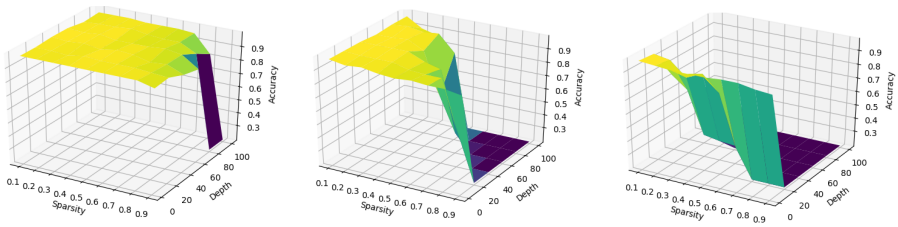
Figure 1: Percentage of weights kept after pruning a randomly initialized FFNN with depth 100 and width 100 for 70% sparsity using SBP on MNIST. Each pixel (i, j) shows the percentage of weights of neuron (i, j) kept after pruning. We can clearly see how the EOC (a) allows us to preserve a uniform spread of the weights, whereas the Chaotic phase (b), due to exploding gradients, prunes entire layers away

or the Chaotic phase. Then the network is ill-conditioned. Moreover, we have

$$\mathbb{E}[s_{cr}] \leq \frac{1}{L}\left(1 + \frac{\log(\kappa L N^2)}{\kappa}\right) + \mathcal{O}\left(\frac{1}{\kappa^2\sqrt{LN^2}}\right),$$

where $\kappa = \frac{|\log(\chi)|}{8}$ *and* $\chi = \sigma_w^2 \mathbb{E}[\phi'(\sqrt{q}Z)^2]$.
Moreover, if (σ_w, σ_b) *are the EOC, then the network is well-conditioned. In this case,* $\kappa = 0$ *and the upper bound no longer holds.*

Theorem 1 shows that initializing on the Ordered or Chaotic phase ($\chi \neq 1$) leads to an upper bound for $\mathbb{E}[s_{cr}]$ of order $\frac{\log(\kappa L N^2)}{\kappa L}$. The farther χ from 1, i.e. the farther the initialization from the EOC, the smaller the upper bound becomes. For constant width FFNN with $L = 100$, $N = 100$ and $\kappa = 1$, the theoretical upper bound is $\mathbb{E}[s_{cr}] \lessapprox 27\%$ while we obtain $\mathbb{E}[s_{cr}] \approx 22\%$ based on 10 simulations. To illustrate the effect of a larger sparsity for the same network, Figure 1 shows the impact of the initialization with sparsity $s = 70\%$. Each pixel represents a neuron in the network and shows the percentage of weights kept after pruning. The dark area in Figure 1(b) shows layers that are fully pruned. This happens because of the exploding gradient on the Chaotic phase. With an initialization on the EOC, Figure 1(a) shows that pruned weights are well distributed in the network, ensuring that no layer is fully pruned.



(a) Tanh with EOC Init & Rescaling (b) Tanh with EOC Init (c) Tanh with Ordered phase Init

Figure 2: Accuracy on MNIST with different initialization schemes including EOC with rescaling, EOC without rescaling, Ordered phase, with varying depth and sparsity. This figure clearly illustrates the benefits of rescaling very sparse and deep FFNN.

3 Training Sparse Networks Using the Rescaling Trick

Training sparse architectures can be very challenging in practice Li et al. [2018]. In our framework, after pruning, the network is no longer on the EOC and the training becomes difficult for deep networks; see, e.g., Schoenholz et al. [2017] and Hayou et al. [2019]. However, by a simple rescaling operation, we show here that we can put the pruned network on the EOC, making it easily trainable.

Consider a FFNN architecture. For two inputs $x, x' \in \mathbb{R}^d$, let $c^l(x, x')$ be the correlation between $y_i^l(x)$ and $y_i^l(x')$ (for some fixed i). It is known from Schoenholz et al. [2017] and Hayou et al. [2019] that there exists a so-called correlation function f such that $c^{l+1}(x, x') = f(c^l(x, x'))$. On the EOC, the hyperparameters (σ_w, σ_b) satisfy the equation $f'(1) = \sigma_w^2 \mathbb{E}[\phi'(\sqrt{q}Z)^2] = 1$ where q is the limiting variance of $y_i^l(x)$ which is usually independent of x and i. After pruning, the forward propagation becomes

$$\hat{y}_i^l(a) = \sum_{j=1}^{N_{l-1}} W_{ij}^l \delta_{ij}^l \phi(\hat{y}_j^{l-1}(a)) + B_i^l, \quad \text{for } l \geq 2, \tag{5}$$

where δ is the pruning mask. This change in the architecture leads to a change in the dynamics of $c^l(x, x')$. Thus, f also changes to become $\hat{f}$ and the equation $\hat{f}'(1) = 1$ is generally not satisfied anymore. Hence, the pruned network is not on the EOC: we say that "pruning destroys the EOC".

One way to address this problem is to re-initialize the pruned network with new weights on the EOC. However, by doing so we lose all information carried by the

Algorithm 1 Rescaling trick for FFNN

Input: Pruned network, size m
for $L = 1$ **to** L **do**
  for $i = 1$ **to** N_l **do**
    $\alpha_i^l \leftarrow \sum_{j=1}^{N_{l-1}} (W_{ij})^2 \delta_{ij}^l$
    $\rho_{ij}^l \leftarrow 1/\sqrt{\alpha_i^l}$ **for all** j
  end for
end for

weights kept after pruning. In Frankle and Carbin [2019], it has been noticed that re-initializing the pruned network leads to poorer performance, which confirms our intuition. Hence, another way to address this problem, with minor changes to the weights, is to introduce scaling factors in the layers. More precisely, we scale the weights in each layer by factors that depend on the pruned architecture itself. This will theoretically ensure that the sparse model is trainable for very deep networks.

Proposition 2 (Rescaling Trick)**.** *Consider a neural network of the form 10 or 11 (FFNN or CNN) initialized on the EOC. Then, after pruning, the sparse network is not initialized on the EOC. However, the rescaled sparse network*

$$y^l(x) = \mathcal{F}(\rho^l \circ \delta^l \circ W^l, y^{l-1}(x)) + B^l, \quad \text{for } l \geq 1, \tag{6}$$

where

- $\rho_{ij}^l = \frac{1}{\sqrt{\mathbb{E}[N_{l-1}(W_{i1}^l)^2 \delta_{i1}^l]}}$ *for FFNN of the form 10,*
- $\rho_{i,j,\beta}^l = \frac{1}{\sqrt{\mathbb{E}[n_{l-1}(W_{i,1,\beta}^l)^2 \delta_{i,1,\beta}^l]}}$ *for CNN of the form 11,*

is initialized on the EOC.

Proposition 7 provides a simple algorithm to put the sparse network on the EOC, therefore making it trainable. The scaling factors are easily approximated using the weights kept after pruning. Algorithm 1 shows a practical implementation of the algorithm for FFNN. Intuitively, by applying the rescaling trick, we ensure that information propagates deeper inside the network as it now lies on the EOC. That way the gradients do not explode or vanish, which makes the sparse model easily trainable. We confirm these results experimentally in Section 6, in particular see Figure 5(d).

4 Pruning Residual Networks

Residual neural networks and their variants He et al. [2015], Huang et al. [2017] are currently the best performing models on various classification tasks (refs, cifar10,

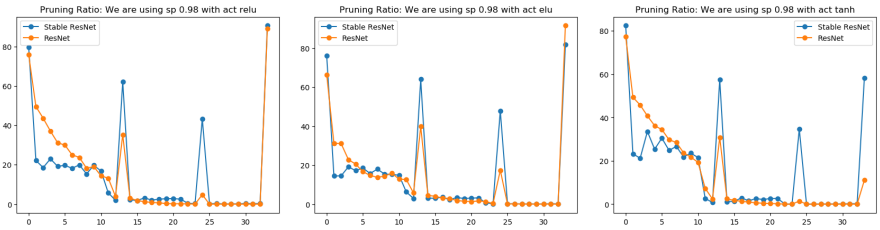


Figure 3: Percentage of pruned weights per layer in a ResNet32 for our Stable ResNet32 and standard Resnet32 with Kaiming initialization on Cifar10. We note that with Stable Renset, we prune less aggressively in the deeper layers than standard Resnet.

cifar100, imagenet etc). Thus, understanding Resnet pruning at initialization is of crucial interest. Results on MBP in section 2.2 apply to Resnet. However, with SBP, results become different. We show here that Resnets are better adapted to pruning at initialization using SBP since they naturally 'live' on the EOC; i.e. no specific initialization is needed. However, Resnets suffer from an exploding gradient problem [Yang and Schoenholz, 2017] which might affect SBP. To address this issue by introducing a new Resnet parameterization. A Resnet architecture is given by

$$\begin{aligned} y^1(x) &= \mathcal{F}(W^1, x), \\ y^l(x) &= y^{l-1}(x) + \mathcal{F}(W^l, y^{l-1}), \quad \text{for } l \geq 2, \end{aligned} \tag{7}$$

where $\mathcal{F}$ defines the blocks of the Resnet. Hereafter, we assume that $\mathcal{F}$ is either of the form (10) or (11) (Fully connected or convolutional layer).

The next theorem shows that Resnet are well-conditioned independently from the initialization and are thus well suited for pruning at initialization.

Theorem 2 (Resnet pruning)**.** *Consider a Resnet with either Fully Connected or Convolutional layers and ReLU activation function. Then for all $\sigma_w > 0$, the Resnet is well-conditioned. Moreover, for all $l \in \{1, ..., L\}, m^l = \Theta((1 + \frac{\sigma_w^2}{2})^L)$.*

Although Resnet are always well-conditioned, the second moment of the pruning criterion grows exponentially in L. This could potentially worsen the pruning and lead to some structural anomalies in the pruned network since the pruning criterion has a big variance. To resolve this situation, we propose a Resnet parameterization which we call Stable Resnet as it stabilizes the pruning criterion.

Proposition 3 (Stable Resnet)**.** *Consider the following Resnet parameterization*

$$y^l(x) = y^{l-1}(x) + \frac{1}{\sqrt{L}}\mathcal{F}(W^l, y^{l-1}), \quad \textit{for } l \geq 2, \tag{8}$$

then the network is well-conditioned for all choices of $\sigma_w > 0$. *Moreover, for all* $l \in \{1, ..., L\}$ *we have* $m^l = \Theta(L^{-1})$.

In Proposition 8, L is the number of residual blocks and not the number of layers. For example ResNet32 has 15 blocks and 32 layers, here $L = 15$. Figure 3 shows the percentage of weights in each layer kept after pruning ResNet32 and Stable ResNet32 at initialization. The jumps correspond to limits between sections in ResNet32 and are caused by max-pooling. Inside each section, Stable Resnet tends to have a more uniform distribution of percentages of weights kept after pruning compared to Standard Resnet.

Unlike Feedforward neural networks (FFNN or CNN), we do not need to rescale the pruned network. Moreover, the next proposition establishes that a Resnet lives on the EOC in the sense that the correlation between $y_i^l(x)$ and $y_i^l(x')$ converges to 1 at a sub-exponential $\mathcal{O}(l^{-2})$ rate.

Proposition 4 (Resnet live on the EOC even after pruning)**.** *Let* x, x' *be two inputs. The following results hold*

1. *For Resnet with Fully Connected layers, let* $\hat{c}^l(x, x')$ *be the correlations between* $\hat{y}_i^l(x)$ *and* $\hat{y}_i^l(x')$ *after pruning the network. Then we have*

$$1 - \hat{c}^l(x, x') \sim \frac{\kappa}{l^2},$$

   where $\kappa > 0$ *is a constant.*

2. *For Resnet with Convolutional layers, let* $\hat{c}^l(x, x') = \frac{\sum_{\alpha,\alpha'} \mathbb{E}[y^l_{1,\alpha}(x) y^l_{1,\alpha'}(x')]}{\sum_{\alpha,\alpha'} \sqrt{\mathbb{E}[y^l_{1,\alpha}(x)^2]} \sqrt{\mathbb{E}[y^l_{1,\alpha}(x')^2]}}$ *be an 'average' correlation after pruning the network. Then we have*

$$1 - \hat{c}^l(x, x') \gtrsim l^{-2}.$$

This is equivalent to what happens when we initialize FFNN or CNN on the EOC and use the Rescaling trick. Resnet networks live naturally on the EOC before and after pruning, thus no rescaling is needed.

5 The Lottery Ticket Hypothesis

Frankle and Carbin [2019] have formulated The Lottery Ticket Hypothesis (LTH) which states that "randomly initialized networks contain subnetworks that when trained in isolation reach test accuracy comparable to the original network". We

have shown so far that pruning a network initialized on the EOC will output sparse architectures that we can train using the rescaling trick. Conversely, if we initialize a random neural network with any hyperparameters (σ_w, σ_b), then intuitively, we can prune this network in a way that ensures that the pruned network is on the EOC. This would theoretically make the sparse architecture trainable. This is established in the next proposition.

Proposition 5 (Winning Tickets on the Edge of Chaos)**.** *Consider a FFNN or CNN with layers initialized with variances $\sigma^2_{w,l} \in \mathbb{R}^+$ for weights and variance $\sigma^2_b > 0$ for bias. Let $\sigma_{w,EOC}$ be the value of σ_w such that $(\sigma_{w,EOC}, \sigma_b) \in EOC$. Then, for all $(\sigma_{w,l})_{l\in[[1,L]]}$ such that $\sigma_{w,l} > \sigma_{w,EOC}$ for all l, there exists a distribution of subnetworks that are initialized on the EOC.*

In Proposition 10, the variances $(\sigma_{w,l})_l$ change from one layer to the other. This makes the result general to any initialization scheme with Gaussian weights. Moreover, proposition 10 establishes the existence of a distribution of subnetworks initialized on the Edge of Chaos. Based on this, We formulate the Generalized Lottery Ticket Hypothesis

Generalized Lottery Ticket Hypothesis: *For any randomly initialized network, there exists a distribution of subnetworks S_n, such that, the average test accuracy achieved by subnetworks drawn from S_n when trained in isolation with the same number of steps, is similar to that of the original network.*

6 Experiments

In this section, we illustrate empirically the theoretical results obtained in the previous sections. We validate the results on MNIST, CIFAR10 and CIFAR100.

6.1 Initialization and rescaling

According to Theorem 1 EOC initialization is necessary for the network to be well-conditioned. We train FFNN with tanh activation on MNIST dataset, varying depth $L \in \{2, 20, 40, 60, 80, 100\}$ and sparsity $s \in \{10\%, 20\%, .., 90\%\}$. We use SGD with batchsize 100 and learning rate 10^{-3}, which we found to be optimal using a grid search with an exponential scale of 10. Figure 5 shows the test accuracy after 10k iterations for 3 different initialization schemes: *Rescaled EOC*, *EOC*, *Ordered*. On the Ordered phase, the model is untrainable when we choose sparsity $s > 40\%$ and depth $L > 60$; this is due to one layer being fully pruned. When we initialize on the EOC, the area of trainable configurations (s, L) becomes larger. However the model is still untrainable

for highly sparse deep networks. The problem here is not structural; nontrainablity is mainly caused by the fact that the sparse network is no longer initialized on the EOC (see proposition 7). In order to get back to the EOC after pruning, we use the rescaling trick. As predicted by Proposition 7, we are able to train the rescaled model appropriately.

Table 2: Classification accuracies for CIFAR10 and CIFAR100 after pruning

	CIFAR10			CIFAR100		
Sparsity	90%	95%	98%	90%	95%	98%
ResNet32 (no pruning)	94.80	-	-	74.64	-	-
OBD LeCun et al. [1990]	93.74	93.58	93.49	73.83	71.98	67.79
Random Pruning	89.95±0.23	89.68±0.15	86.13±0.25	63.13±2.94	64.55±0.32	19.83±3.21
MBP	90.21±0.55	88.35±0.75	86.83±0.27	67.07±0.31	64.92±0.77	59.53±2.19
SNIP Lee et al. [2018a]	92.26 ± 0.32	91.18 ± 0.17	87.78 ± 0.16	69.31 ± 0.52	65.63 ± 0.15	55.70 ± 1.13
GraSP Wang et al. [2020]	92.20±0.31	**91.39±0.25**	**88.70±0.42**	69.24 ± 0.24	66.50 ± 0.11	58.43 ± 0.43
GraSP-SR	91.95±0.22	91.16±0.13	87.8 ± 0.32	69.12 ± 0.15	65.49 ± 0.21	58.0 ± 0.18
SBP-SR (Stable ResNet)	**92.56 ± 0.06**	91.21 ± 0.30	88.25 ± 0.35	**69.51 ± 0.21**	**66.72 ± 0.12**	**59.51 ± 0.15**
ResNet50 (no pruning)	94.90	-	-	74.9	-	-
Random Pruning	85.11±4.51	88.76±0.21	85.32±0.47	65.67±0.57	60.23±2.21	28.32±10.35
MBP	90.11 ± 0.32	89.06 ± 0.09	87.32 ± 0.16	68.51 ± 0.21	63.32 ± 1.32	55.21 ± 0.35
SNIP	91.95 ± 0.13	92.12 ± 0.34	89.26 ± 0.23	70.43 ± 0.43	67.85 ± 1.02	60.38 ± 0.78
GraSP	**92.10 ± 0.21**	91.74 ± 0.35	**89.97± 0.25**	70.53±0.32	67.84±0.25	63.88±0.45
SBP-SR	92.05 ± 0.06	**92.74± 0.32**	89.57 ± 0.21	**71.79 ± 0.13**	**68.98 ± 0.15**	**64.45 ± 0.34**
ResNet104 (no pruning)	94.92	-	-	75.24	-	-
Random Pruning	89.80±0.33	87.86±1.22	85.52±2.12	66.73±1.32	64.98±0.11	30.31±4.51
MBP	90.05 ± 1.23	88.95±0.65	87.83±1.21	69.57±0.35	64.31±0.78	60.21±2.41
SNIP	93.25 ± 0.53	92.98 ± 0.12	91.58 ± 0.19	71.94 ± 0.22	68.73±0.09	63.31 ± 0.41
GraSP	93.08 ± 0.17	92.93 ± 0.09	91.19±0.35	73.33±0.21	70.95 ± 1.12	66.91±0.33
SBP-SR	**93.90 ± 0.13**	**93.88 ± 0.17**	**92.08 ± 0.14**	**74.17 ± 0.11**	**71.84 ± 0.13**	**67.73 ± 0.28**

6.2 Resnet and Stable Resnet

Although Resnets are naturally adapted to pruning with SBP (i.e. they are always well-conditioned for all $\sigma_w > 0$), Theorem 4 shows that the magnitude of the pruning criterion grows exponentially with respect to the depth L. To resolve this problem we introduced Stable Resnet. We call our pruning algorithm for ResNet SBP-SR (Sensitivity Based Pruning with Stable Resnet). Theoretically, we expect SBP-SR to perform better than other methods for deep Resnets according to Proposition 8. Table 2 shows test accuracies for ResNet32, ResNet50 and ResNet104 with varying sparsities $s \in \{90\%, 95\%, 98\%\}$ on CIFAR10 and CIFAR100. For all our experiments, we use a setup similar to [Wang et al., 2020], i.e. we use SGD for 160 and 250 epochs for CIFAR10 and CIFAR100 respectively. We use an initial learning rate of 0.1 and decay the learning rate by 0.1 at 1/2 and 3/4 of the number of total epoch. In addition,

we run all our experiments 3 times so as to obtain more stable and trustworthy test accuracies. In addition, just like in [Wang et al., 2020], we adopt Resnet architectures where we doubled the number of filters in each convolutional layer.

As a baseline, we include pruning results with the OBD algorithm LeCun et al. [1990] for ResNet32 which is a classical pruning algorithm (train $\rightarrow$ prune $\rightarrow$ repeat). We compare our results against other algorithms that prune at initialization, such as SNIP Lee et al. [2018a], which is a SBP algorithm, and GraSP Wang et al. [2020] which is a Hessian based pruning algorithm. SBP-SR outperforms other algorithms that prune at initialization, in deep networks (ResNet104). Furthermore, we note that on all Cifar100 experiments, SBP-SR also performs significantly better than other one-shot pruning algorithms. Using GraSP on Stable Resnet did not improve the result of GraSP on standard Resnet, as our proposed Stable Resnet analysis only applies to gradient based pruning. The analysis of Hessian based pruning could lead to similar techniques for improving trainability, which we leave for future work.

Table 3 shows a stress-test of the SBP-SR with very high sparsities $s \in \{99.5\%, 99.9\%\}$. For 99.9% sparsity, we still get 72.70% test accuracy with ResNet104 whereas, with SNIP on standard ResNet104, the model is non trainable and stuck at the random classifier accuracy of 10%.

Table 3: Classification accuracies on CIFAR10 for Resnet with varying depths

	ALGORITHM	99.5%	99.9%
RESNET32	SNIP	77.56±0.36	9.98±0.08
	SBP-SR	**79.54±1.12**	**51.56±1.12**
RESNET50	SNIP	80.49±2.41	19.98±14.12
	SBP-SR	**82.68±0.52**	**58.76±1.82**
RESNET104	SNIP	33.63±33.27	10.11±0.09
	SBP-SR	**87.47±0.23**	**72.70±0.48**

7 Conclusion

In this paper, we have formulated principled guidelines for pruning at initialization. We have derived bounds for the maximal sparsity one can achieve without having at least one layer fully pruned, and have introduced a rescaling trick to make the pruned network trainable. We have also shown why Resnets are well suited for pruning and introduced a new Resnet parameterization called Stable Resnet, which allows

for more stable pruning. Our theoretical results on MNIST, Cifar10 and Cifar100 have been validated by extensive experiments. Compared to other available one-shot pruning algorithms, we achieve state-of the-art results for very deep networks such as ResNet104. In addition, we have also stress-tested the SBP-SR pruning algorithm and show how one can prune up to 99.9% of the weights while still remaining trainable. Lastly, we have demonstrated theoretical results that support the Lottery Ticket Hypothesis and generalized it.

References

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

B. Hassibi, D. G.Stork, and W. Gregory. Optimal brain surgeon and general network pruning. In *IEEE International Conference on Neural Networks*, pages 293 – 299 vol.1, 02 1993.

N. Lee, T. Ajanthan, and P HS. Torr. Snip: Single-shot network pruning based on connection sensitivity. In *ICLR*, 2018a.

Q. Nguyen and M. Hein. Optimization landscape and expressivity of deep CNNs. In *ICML*, 2018.

S.S. Du, X. Zhai, B. Poczos, and A. Singh. Gradient descent provably optimizes over-parameterized neural networks. In *ICLR*, 2019.

C. Zhang, S. Bengio, M. Hardt, B. Recht, and O. Vinyals. Understanding deep learning requires rethinking generalization. In *ICLR*, 2016.

B. Neyshabur, Z. Li, S. Bhojanapalli, Y. LeCun, and N. Srebro. The role of over-parametrization in generalization of neural networks. In *ICLR*, 2019.

Michael C Mozer and Paul Smolensky. Skeletonization: A technique for trimming the fat from a network via relevance assessment. In *Advances in Neural Information Processing Systems*, pages 107–115, 1989.

X. Dong, S. Chen, and S.J. Pan. Learning to prune deep neural networks via layer-wise optimal brain surgeon. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, page 4860–4874, 2017.

C. Louizos, M. Welling, and D.P. Kingma. Learning sparse neural networks through l_0 regularization. In *ICLR*, 2018.

M.Á. Carreira-Perpiñán and Y. Idelbayev. Learning-compression algorithms for neural net pruning. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

J. Frankle and M. Carbin. The lottery ticket hypothesis: Finding sparse, trainable neural networks. In *ICLR*, 2019.

C. Wang, G. Zhang, and R. Grosse. Picking winning tickets before training by preserving gradient flow. In *ICLR*, 2020.

H. Li, A. Kadav, I. Durdanovic, H. Samet, and H.P. Graf. Pruning filters for efficient convnets. In *ICLR 2018*, 2018.

S. Hayou, A. Doucet, and J. Rousseau. On the impact of the activation function on deep neural networks training. In *ICML*, 2019.

S.S. Schoenholz, J. Gilmer, S. Ganguli, and J. Sohl-Dickstein. Deep information propagation. In *5th International Conference on Learning Representations*, 2017.

B. Poole, S. Lahiri, M. Raghu, J. Sohl-Dickstein, and S. Ganguli. Exponential expressivity in deep neural networks through transient chaos. In *30th Conference on Neural Information Processing Systems*, 2016.

G. Yang and S. Schoenholz. In *Mean Field Residual Networks: On the Edge of Chaos*, volume 30, pages 2869–2869, 2017.

L. Xiao, Y. Bahri, J. Sohl-Dickstein, S. S. Schoenholz, and P. Pennington. Dynamical isometry and a mean field theory of cnns: How to train 10,000-layer vanilla convolutional neural networks. In *ICML 2018*, 2018.

J. Lee, Y. Bahri, R. Novak, S.S. Schoenholz, J. Pennington, and J. Sohl-Dickstein. Deep neural networks as Gaussian processes. In *6th International Conference on Learning Representations*, 2018b.

A.G. Matthews, J. Hron, M. Rowland, R.E. Turner, and Z. Ghahramani. Gaussian process behaviour in wide deep neural networks. In *6th International Conference on Learning Representations*, 2018.

K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2015.

G. Huang, Z. Liu, L.V.D Maaten, and K.Q. Weinberger. Densely connected convolutional networks. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2261–2269, 2017.

G. Yang. Scaling limits of wide neural networks with weight sharing: Gaussian process behavior, gradient independence, and neural tangent kernel derivation. *arXiv preprint arXiv:1902.04760*, 2019.

M.L. Puri and S.S. Ralescu. Limit theorems for random central order statistics. *Lecture Notes-Monograph Series Vol. 8, Adaptive Statistical Procedures and Related Topics (1986), pp. 447-475*, 1986.

G.H. Hardy, J.E. Littlewood, and G. Pólya. *Inequalities*, volume 2. Cambridge Mathematical Library, 1952.

We provide in Sections B, C and D the proofs of the theoretical results presented in the main document. Section G provides additional empirical results. Hereafter, "Appendix Lemma" and "Appendix Proposition" refer to results that are in the appendix but not in the main paper.

A Preliminary results

Let x be an input in $\mathbb{R}^d$. In its general form, a neural network of depth L is given by the following set of forward propagation equations

$$y^l(x) = \mathcal{F}_l(W^l, y^{l-1}(x)) + B^l, \quad 1 \leq l \leq L \tag{9}$$

where $y^l(x)$ is the vector of pre-activations and W^l and B^l are respectively the weights and bias of the l^{th} layer. F_l is a mapping that defines the nature of the layer. The weights and bias are initialized with $W^l \overset{iid}{\sim} \mathcal{N}(0, \frac{\sigma_w^2}{v_l})$ where v_l is a scaling factor used to control the variance of y^l, and $B^l \overset{iid}{\sim} \mathcal{N}(0, \sigma_b^2)$. Hereafter, we denote by M_l the number of weights in the l^{th} layer, ϕ the activation function and $[|n, m|]$ the set of integers $\{n, n+1, ..., m\}$ for $n \leq m$. Two examples of such architectures are

- **Fully-connected FeedForward Neural Network (FFNN)**
  For a fully connected feedforward neural network of depth L and widths $(N_l)_{0 \leq l \leq L}$, the forward propagation of the input through the network is given by

$$\begin{aligned} y_i^1(x) &= \sum_{j=1}^{d} W_{ij}^1 x_j + B_i^1, \\ y_i^l(x) &= \sum_{j=1}^{N_{l-1}} W_{ij}^l \phi(y_j^{l-1}(x)) + B_i^l, \quad \text{for } l \geq 2. \end{aligned} \tag{10}$$

  Here, we have $v_l = N_{l-1}$ and $M_l = N_{l-1}N_l$.

- **Convolutional Neural Network (CNN/ConvNet)**
  For a 1D convolutional neural network of depth L, number of channels $(n_l)_{l \leq L}$ and number of neurons per channel $(N_l)_{l \leq L}$. we have

$$\begin{aligned} y_{i,\alpha}^1(x) &= \sum_{j=1}^{n_{l-1}} \sum_{\beta \in ker_l} W_{i,j,\beta}^1 x_{j,\alpha+\beta} + b_i^1, \\ y_{i,\alpha}^l(x) &= \sum_{j=1}^{n_{l-1}} \sum_{\beta \in ker_l} W_{i,j,\beta}^l \phi(y_{j,\alpha+\beta}^{l-1}(x)) + b_i^l, \text{ for } l \geq 2, \end{aligned} \tag{11}$$

where $i \in [|1, n_l|]$ is the channel index, $\alpha \in [|0, N_l - 1|]$ is the neuron location, $ker_l = [|-k_l, k_l|]$ is the filter range and $2k_l + 1$ is the filter size. To simplify the analysis, we assume hereafter that $N_l = N$ and $k_l = k$ for all l. Here, we have $v_l = n_{l-1}(2k + 1)$ and $M_l = n_{l-1}n_l(2k + 1)$. We assume periodic boundary conditions, so $y^l_{i,\alpha} = y^l_{i,\alpha+N} = y^l_{i,\alpha-N}$. Generalization to multidimensional convolutions is straighforward.

We start by recalling some results from the Mean Field Theory of Neural Nets.

Edge of Chaos (EOC): For some input x, we denote by $q^l(x)$ the variance of $y^l(x)$. The convergence of $q^l(x)$ as l increases has been studied in Schoenholz et al. [2017] and Hayou et al. [2019]. In particular, under weak regularity conditions, they prove that $q^l(x)$ converges to a point $q(\sigma_b, \sigma_w) > 0$ independent of x as $l \to \infty$. The asymptotic behaviour of the correlations $c^l(x, x')$ between $y^l(x)$ and $y^l(x')$ for any two inputs x and x' is also driven by (σ_b, σ_w): the dynamics of c^l are controlled by a function f i.e. $c^{l+1} = f(c^l)$ called the correlation function. The authors define the EOC as the set of parameters (σ_b, σ_w) such that $\sigma_w^2 \mathbb{E}[\phi'(\sqrt{q(\sigma_b, \sigma_w)}Z)^2] = 1$ where $Z \sim \mathcal{N}(0, 1)$. Similarly the Ordered, resp. Chaotic, phase is defined by $\sigma_w^2 \mathbb{E}[\phi'(\sqrt{q(\sigma_b, \sigma_w)}Z)^2] < 1$, resp. $\sigma_w^2 \mathbb{E}[\phi'(\sqrt{q(\sigma_b, \sigma_w)}Z)^2] > 1$. On the Ordered phase, the gradient will vanish as it backpropagates through the network, and the correlation $c^l(x, x')$ converges exponentially to 1. Hence the output function becomes constant (hence the name 'Ordered phase'). On the Chaotic phase, the gradient explodes and the correlation converges exponentially to some limiting value $c < 1$ which results in the output function being discontinuous everywhere (hence the 'Chaotic' phase name). On the EOC, the second moment of the gradient remains constant throughout the backpropagation and the correlation converges to 1 at a sub-exponential rate, which allows deeper information propagation.

We also have $f'(1) = \sigma_w^2 \mathbb{E}[\phi'(\sqrt{q(\sigma_b, \sigma_w)}Z)^2]$, which means the EOC is also defined by $f'(1) = 1$. In the limit of infinitely wide FFNN, we have the following results (Hayou et al. [2019]) :

- There exist $q, \lambda > 0$ such that, for all $\sup_{x \in \mathbb{R}^d} |q^l - q| \leq e^{-\lambda l}$.
- On the Ordered phase, there exists $\gamma > 0$ such that $\sup_{x,x' \in \mathbb{R}^d} |c^l(x, x') - 1| \leq e^{-\gamma l}$.
- On the Chaotic phase, there exist $\gamma > 0$ and $c < 1$ such that $\sup_{x \neq x' \in \mathbb{R}^d} |c^l(x, x') - c| \leq e^{-\gamma l}$.

- For ReLU network on the EOC, we have

$$f(x) \underset{x\to 1-}{=} x + \frac{2\sqrt{2}}{3\pi}(1-x)^{3/2} + O((1-x)^{5/2}).$$

- In general, we have

$$f(x) = \frac{\sigma_b^2 + \sigma_w^2 \mathbb{E}[\phi(\sqrt{q}Z_1)\phi(\sqrt{q}Z(x))]}{q},$$

  where $Z(x) = xZ_1 + \sqrt{1-x^2}Z_2$ and Z_1, Z_2 are iid standard Gaussian variables.

- On the EOC, we have $f'(1) = 1$

- For non-linear activation functions, f is strictly convex and $f(1) = 1$.

Similar results exist for CNN. Xiao et al. [2018] studied the limiting behaviour of correlations $c^l_{\alpha,\alpha'}(x,x)$ (same input x). These correlations describe how features are correlated for the same input. However, they do not capture the behaviour of these features for different inputs (ie $c^l_{\alpha,\alpha'}(x,x')$ where $x \neq x'$). We establish this result here.

Appendix Lemma 1 (Asymptotic behaviour of the correlation in CNN with smooth activation functions)**.** *We consider a 1D CNN. Let $(\sigma_b, \sigma_w) \in (\mathbb{R}^+)^2$ and x, x' be two inputs. If (σ_b, σ_w) are either on the Ordered or Chaotic phase, then there exists $\beta > 0$ such that*

$$\sup_{\alpha,\alpha'} |c^l_{\alpha,\alpha'}(x,x') - c| = \mathcal{O}(e^{-\beta l}),$$

where $c = 1$ if (σ_b, σ_w) is in the Ordered phase, and $c \in (0,1)$ if (σ_b, σ_w) is in the Chaotic phase.

Proof. Let x, x' be two inputs and α, α' two nodes in the same channel i. Using the central limit theorem in the large c (number of channels) limit, we have

$$q^l_{\alpha,\alpha'}(x,x') = \mathbb{E}[y^l_{i,\alpha}(x) y^l_{i,\alpha'}(x')] = \frac{\sigma_w^2}{2k+1} \sum_{\beta \in ker} \mathbb{E}[\phi(y^{l-1}_{1,\alpha+\beta}(x))\phi(y^{l-1}_{1,\alpha'+\beta}(x'))] + \sigma_b^2.$$

This yields

$$c^l_{\alpha,\alpha'}(x,x') = \frac{1}{2k+1} \sum_{\beta \in ker} f(c^{l-1}_{\alpha+\beta,\alpha'+\beta}(x,x')).$$

We present the Ordered phase, the proof in the Chaotic phase is similar. Let (σ_b, σ_w) be in the Ordered phase and $c^l_m = \min_{\alpha,\alpha'} c^l_{\alpha,alpha'}(x,x')$. Using the fact that f is non-decreasing, we have that $c^l_{\alpha,\alpha'}(x,x') \geq \frac{1}{2k+1} \sum_{\beta \in ker} c^{l-1}_{\alpha+\beta,\alpha'+\beta}(x,x')) \geq f(c^{l-1}_m)$. Taking

the min again over α, α', we have $c_m^l \geq f(c_m^{l-1})$, therefore c_m^l is non-decreasing and converges to a stable fixed point of f. By the convexity of f, the limit is 1 (in the Chaotic phase, f has two fixed point, a stable point $c_1 < 1$ and $c_2 = 1$ unstable). Moreover, the convergence is exponential using the fact that $0 < f'(1) < 1$.

□

Gradient Independence : Yang [2019] has shown that "in the mean field approximation, assuming that the weights used for forward propagation are independent of those used for backpropagation for usual architectures, leads to correct calculation for gradient backpropagation". We use this result in our proofs.

B Proofs for Section 2 : Neural Networks Pruning

Proposition 6 (MBP in the large depth limit)**.** *Assume there exists $l_0 \in [|1, L|]$ such that $\alpha_{l_0} > \alpha_l$ for all l, and let Q_x be the x quantile of the folded standard normal distribution for $x \in [0, 1]$ (i.e. Q_x are quantiles of $|X|$ where $X \underset{iid}{\sim} \mathcal{N}(0, 1)$). Let $\gamma = \min_{l \neq l_0} \frac{\alpha_{l_0}}{\alpha_l}$, and, for $\epsilon \in (0, 2)$, define $x_{\epsilon,\gamma} = \min\{y \in (0, 1) : \forall x > y, \gamma Q_x > Q_{1-(1-x)^{\gamma^{2-\epsilon}}}\}$ if the set is not null and $x_{\epsilon,\gamma} = \infty$ otherwise. Then, for all $\epsilon \in (0, 2)$, $x_{\epsilon,\gamma}$ is finite, and there exists a constant $\nu > 0$ such that*

$$\mathbb{E}[s_{cr}] \leq \inf_{\epsilon \in (0,2)} \{x_{\epsilon,\gamma} + \frac{\zeta_{l_0} N^2}{1 + \gamma^{2-\epsilon}} (1 - x_{\epsilon,\gamma})^{1+\gamma^{2-\epsilon}}\} + \frac{\nu}{\sqrt{LN^2}}.$$

Proof. Let $x \in (0, 1)$ and $k_x = (1 - x)\Gamma_L N^2$, where $\Gamma_L = \sum_{l \neq l_0} \zeta_l$. We have that

$$\mathbb{P}(s_{cr} \leq x) \geq \mathbb{P}(\max_i |W_i^{l_0}| < |W|^{(k_x)}),$$

where $|W|^{(k_x)}$ is the k_x^{th} order statistic of the sequence $\{|W_i^l|, l \neq l_0, i \in [|1, M_l|]\}$; i.e $|W|^{(1)} > |W|^{(2)} > ... > |W|^{(k_x)}$.

Let $(X_i)_{i \in [|1, M_{l_0}|]}$ and $(Z_i)_{i \in [|1, \Gamma_L N^2|]}$ be two sequences of iid standard normal variables. It is easy to see that

$$\mathbb{P}(\max_{i,j} |W_{ij}^{l_0}| < |W|^{(k_x)}) \geq \mathbb{P}(\max_i |X_i| < \gamma |Z|^{(k_x)})$$

where $\gamma = \min_{l \neq l_0} \frac{\alpha_{l_0}}{\alpha_l}$.

Moreover, we have the following result from Order Statistics Theory

Appendix Lemma 2. *Let $X_1, X_2, ..., X_n$ be iid random variables with a cdf F. Assume F is differentiable and let $p \in (0,1)$ and let Q_p be the order p quantile of the distribution F i.e. $F(Q_p) = p$. Then we have*

$$\sqrt{n}(X^{(pn)} - Q_p)F'(Q_p)\sigma_p^{-1} \underset{D}{\to} \mathcal{N}(0,1),$$

where the convergence is in distribution and $\sigma_p = p(1-p)$.

Appendix lemma 2 is a weak version of a general result detailed in Theorem 3.1. in Puri and Ralescu [1986]. Using this result, we obtain

$$\mathbb{P}(\max_i |X_i| < \gamma |Z|^{(k_x)}) = \mathbb{P}(\max_i |X_i| < \gamma Q_x) + \mathcal{O}(\frac{1}{\sqrt{LN^2}}),$$

where Q_x is the x quantile of the folded standard normal distribution.

The next result shows that $x_{\epsilon,\gamma}$ is finite for all $\epsilon \in (0,2)$.

Appendix Lemma 3. *For all $\epsilon \in (0,2)$, there exists $x_\epsilon \in (0,1)$ such that, for all $x > x_\epsilon$, $\gamma Q_x > Q_{1-(1-x)^{\gamma^{2-\epsilon}}}$.*

Proof. Let $\epsilon > 0$, and recall the asymptotic equivalent of Q_{1-x} given by

$$Q_{1-x} \sim_{x \to 0} \sqrt{-2\log(x)}$$

Therefore, $\frac{\gamma Q_x}{Q_{1-(1-x)^{\gamma^{2-\epsilon}}}} \sim_{x \to 1} \sqrt{\gamma^\epsilon} > 1$. Hence x_ϵ exists and is finite. □

Let $\epsilon > 0$. Using Lemma 3, there exists $x_\epsilon > 0$ such that

$$\begin{aligned}
\mathbb{P}(\max_i |X_i| < \gamma Q_x) &\geq \mathbb{P}(\max_i |X_i| < Q_{1-(1-x)^{\gamma^{2-\epsilon}}}) \\
&= (1 - (1-x)^{\gamma^{2-\epsilon}})^{\zeta_{l_0} N^2} \\
&\geq 1 - \zeta_{l_0} N^2 (1-x)^{\gamma^{2-\epsilon}},
\end{aligned}$$

where we have used the inequality $(1-t)^z \geq 1 - zt$ for all $(t,z) \in [0,1] \times (1,\infty)$ and $\beta = \alpha_{l_0}\alpha_{l_0+1}$.

Using the last result, we have

$$\mathbb{P}(s_{cr} \geq x) \leq \beta N^2 (1-x)^{\gamma^{2-\epsilon}} + \mathcal{O}(\frac{1}{\sqrt{LN^2}}).$$

Now we have

$$\begin{aligned}\mathbb{E}[s_{cr}] &= \int_0^1 \mathbb{P}(s_{cr} \geq x)dx \\ &\leq x_\epsilon + \int_{x_\epsilon}^1 \mathbb{P}(s_{cr} \geq x)dx \\ &\leq x_\epsilon + \frac{\beta N^2}{1+\gamma^{2-\epsilon}}(1-x_\epsilon)^{\gamma^{2-\epsilon}+1} + \mathcal{O}(\frac{1}{\sqrt{LN^2}}).\end{aligned}$$

This is true for all $\epsilon \in (0,2)$, and the additional term $\mathcal{O}(\frac{1}{\sqrt{LN^2}})$ does not depend on ϵ. Therefore there exists a constant $\nu \in \mathbb{R}$ such that for all ϵ

$$\mathbb{E}[s_{cr}] \leq x_\epsilon + \frac{\beta N^2}{1+\gamma^{2-\epsilon}}(1-x_\epsilon)^{\gamma^{2-\epsilon}+1} + \frac{\nu}{\sqrt{LN^2}}.$$

we conclude by taking the infimum over ϵ. □

Theorem 3 (Initialization is crucial for SBP)**.** *We consider a neural network of type 10 or 11 (FFNN or CNN). Assume (σ_w, σ_b) are chosen to be either in the Ordered or the Chaotic phase. Then the network is ill-conditioned. Moreover, we have*

$$\mathbb{E}[s_{cr}] \leq \frac{1}{L}(1 + \frac{\log(\kappa LN^2)}{\kappa}) + \mathcal{O}(\frac{1}{\kappa^2\sqrt{LN^2}})$$

where $\kappa = \frac{|\log(\chi)|}{8}$ and $\chi = \sigma_w^2 \mathbb{E}[\phi'(\sqrt{q}Z)^2]$.
Moreover, if (σ_w, σ_b) are chosen to be on the EOC, then the network is well-conditioned. In this case, $\kappa = 0$ and the upper bound no longer holds.

Proof. We prove the result for the Ordered phase, the proof for the Chaotic phase is similar.

1. **Case 1 : Fully connected Feedforward Neural Networks**

   To simplify the proof, we assume that $M_l = N^2$ for all l. Generalization to other cases is straightforward.
   Let $\epsilon > 0$, and $x > \frac{1}{L} + \epsilon$. With sparsity x, we keep $K_x = (1-x)LN^2$ weights. We have that

   $$\mathbb{P}(s_{cr} \leq x) \geq \mathbb{P}(\max_{i,j} |W_{ij}^0| \Big| \frac{\partial \mathcal{L}}{\partial W_{ij}^0} \Big| < t^{(k_x)})$$

   where $t^{(k_x)}$ is the k_x^{th} order statistic of the sequence $\{|W_{ij}^l| \Big| \frac{\partial \mathcal{L}}{\partial W_{ij}^l} \Big|, l > 0, (i,j) \in [|1, N_l|] \times [|1, N_{l-1}|]\}$.

We have that

$$\frac{\partial \mathcal{L}}{\partial W_{ij}^l} = \frac{1}{|\mathcal{D}|} \sum_{x \in \mathcal{D}} \frac{\partial \mathcal{L}}{\partial y_i^l(x)} \frac{\partial y_i^l(x)}{\partial W_{ij}^l}$$
$$= \frac{1}{|\mathcal{D}|} \sum_{x \in \mathcal{D}} \frac{\partial \mathcal{L}}{\partial y_i^l(x)} \phi(y_j^{l-1}(x)).$$

On the Ordered/Chaotic phase, the variance q^l, the correlation c^l, and the correlation of the gradients $\tilde{c}^l$ converge exponentially to their limiting values $q, 1$ and 0 respectively. To simplify the proof, we use the following approximations (the result holds true without using these approximations, but the full proof requires many unnecessary complications):

- $\forall x \neq x', c_{xx'}^l \approx 1$
- $\forall x, q_{xx}^l \approx q$

using these approximations, we have that $y_i^l(x) = y_i^l(x')$ almost surely for all x, x'. Thus

$$\mathbb{E}\Big[\frac{\partial \mathcal{L}}{\partial W_{ij}^l}^2\Big] = \mathbb{E}[\phi(\sqrt{q}Z)^2]\tilde{q}_x^l,$$

where x is an input. The choice of x is not important in our approximation. The backpropagation of the gradient is given by the set of equations

$$\frac{\partial \mathcal{L}}{\partial y_i^l} = \phi'(y_i^l) \sum_{j=1}^{N_{l+1}} \frac{\partial \mathcal{L}}{\partial y_j^{l+1}} W_{ji}^{l+1}.$$

Using the approximation that the weights used for forward propagation are independent from those used in backpropagation, we have that

$$\tilde{q}_x^l = \tilde{q}_x^{l+1} \frac{N_{l+1}}{N_l} \chi.$$

Then we obtain

$$\tilde{q}_x^l = \frac{N_L}{N_l} \tilde{q}_x^L \chi^{L-l},$$

where $\chi = \sigma_w^2 \mathbb{E}[\phi(\sqrt{q}Z)^2]$. Without loss of generality, we can assume the widths are equal, i.e. $N_1 = N_2 = \cdots = N_L$, we have that

$$\tilde{q}_x^l = \tilde{q}_x^L \chi^{L-l}.$$

Note that by definition, one has $\chi < 1$ on the Ordered phase. Using this result, we have

$$\mathbb{E}\Big[\frac{\partial \mathcal{L}}{\partial W_{ij}^l}^2\Big] = A\ \chi^{L-l},$$

where $A = \mathbb{E}[\phi(\sqrt{q}Z)^2]\tilde{q}_x^L$ for an input x. Note that in the general case where the widths are different, $\tilde{q}^l$ will also scale as χ^{L-l} up to a different constant. Now we want to lower bound the probability

$$\mathbb{P}(\max_{i,j} |W_{ij}^1| \Big| \frac{\partial \mathcal{L}}{\partial W_{ij}^0}\Big| < t^{(k_x)}).$$

Let $t_\epsilon^{(K_x)}$ be the k_x^{th} order statistic of the sequence $\{|W_{ij}^l| \Big| \frac{\partial \mathcal{L}}{\partial W_{ij}^l}\Big|, l > 1+\epsilon L, (i,j) \in [|1, N_l|] \times [|1, N_{l-1}|]\}$. It is clear that $t^{(k_x)} > t_\epsilon^{(k_x)}$, therefore

$$\mathbb{P}(\max_{i,j} |W_{ij}^1| \Big| \frac{\partial \mathcal{L}}{\partial W_{ij}^1}\Big| < t^{(k_x)}) \geq \mathbb{P}(\max_{i,j} |W_{ij}^1| \Big| \frac{\partial \mathcal{L}}{\partial W_{ij}^1}\Big| < t_\epsilon^{(k_x)}).$$

Using Markov's inequality, we have that

$$\mathbb{P}(\Big| \frac{\partial \mathcal{L}}{\partial W_{ij}^0}\Big| \geq \alpha) \leq \frac{\mathbb{E}\Big[\Big| \frac{\partial \mathcal{L}}{\partial W_{ij}^1}\Big|^2\Big]}{\alpha^2}. \tag{12}$$

Note that $Var(\chi^{\frac{l-L}{2}} \Big| \frac{\partial \mathcal{L}}{\partial W_{ij}^l}\Big|) = A$. Assume that the random variables $\chi^{\frac{l-L}{2}} \Big| \frac{\partial \mathcal{L}}{\partial W_{ij}^l}\Big|$ have a density f_{ij}^l for all $l > 1 + \epsilon L, (i,j) \in [|1, N_l|] \times [|1, N_{l-1}|]$, such that $f_{ij}^l(0) \neq 0$. Therefore, there exists a constant λ such that for x small enough,

$$\mathbb{P}(\chi^{\frac{l-L}{2}} \Big| \frac{\partial \mathcal{L}}{\partial W_{ij}^l}\Big| \geq x) \geq 1 - \lambda x.$$

By selecting $x = \chi^{\frac{(1-\epsilon/2)L-1}{2}}$, we have that

$$\chi^{\frac{l-L}{2}} \times x \leq \chi^{\frac{(1+\epsilon L)-L}{2}} \chi^{\frac{(1-\epsilon/2)L-1}{2}} = \chi^{\epsilon L/2}.$$

Therefore, for L large enough, and all $l > 1 + \epsilon L$, $(i,j) \in [|1, N_l|] \times [|1, N_{l-1}|]$, we have that

$$\mathbb{P}(\Big| \frac{\partial \mathcal{L}}{\partial W_{ij}^l}\Big| \geq \chi^{\frac{(1-\epsilon/2)L-1}{2}}) \geq 1 - \lambda\ \chi^{\frac{l-(\epsilon L/2+1)}{2}} \geq 1 - \lambda\ \chi^{\epsilon L/2}.$$

Now choosing $\alpha = \chi^{\frac{(1-\epsilon/4)L-1}{2}}$ in inequality (12) yields

$$\mathbb{P}(\Big|\frac{\partial \mathcal{L}}{\partial W_{ij}^1}\Big| \geq \chi^{\frac{(1-\epsilon/4)L-1}{2}}) \geq 1 - A\ \chi^{\epsilon L/4}.$$

Since we do not know the exact distribution of the gradients, the trick is to bound them using the previous concentration inequalities. We define the event $B := \{\forall (i,j) \in [|1, N_1|] \times [|1, N_0|], \Big|\frac{\partial \mathcal{L}}{\partial W_{ij}^1}\Big| \leq \chi^{\frac{(1-\epsilon/4)L-1}{2}}\} \cap \{\forall l > 1 + \epsilon L, (i,j) \in [|1, N_l|] \times [|1, N_{l-1}|], \Big|\frac{\partial \mathcal{L}}{\partial W_{ij}^l}\Big| \geq \chi^{\frac{(1-\epsilon/2)L-1}{2}}\}$.

We have that

$$\mathbb{P}(\max_{i,j} |W_{ij}^1| \Big|\frac{\partial \mathcal{L}}{\partial W_{ij}^1}\Big| < t_\epsilon^{(k_x)}) \geq \mathbb{P}(\max_{i,j} |W_{ij}^1| \Big|\frac{\partial \mathcal{L}}{\partial W_{ij}^1}\Big| < t_\epsilon^{(k_x)} \Big| B)\mathbb{P}(B).$$

But, by conditioning on the event B, we have

$$\mathbb{P}(\max_{i,j} |W_{ij}^1| \Big|\frac{\partial \mathcal{L}}{\partial W_{ij}^1}\Big| < t_\epsilon^{(k_x)} \Big| B) \geq \mathbb{P}(\max_{i,j} |W_{ij}^1| < \chi^{-\epsilon L/8} {t'}_\epsilon^{(k_x)}),$$

where ${t'}_\epsilon^{(k_x)}$ is the k_x^{th} order statistic of the sequence $\{|W_{ij}^l|, l > 1 + \epsilon L, (i,j) \in [|1, N_l|] \times [|1, N_{l-1}|]\}$.

Now, as in the previous proof, define $x_{\zeta,\gamma_L} = \min\{y \in (0,1) : \forall x > y, \gamma_L Q_x > Q_{1-(1-x)^{\gamma_L^{2-\zeta}}}\}$, where $\gamma_L = \chi^{-\epsilon L/8}$. Since $\lim_{\zeta \to 2} x_{\zeta,\gamma_L} = 0$, then there exists $\zeta_\epsilon = 2$ such that $x_{\zeta_\epsilon,\gamma_L} < \epsilon + \frac{1}{L}$.

As L grows, ${t'}_\epsilon^{(k_x)}$ converges to the quantile of order $\frac{x-\epsilon}{1-\epsilon}$. Therefore,

$$\begin{aligned}\mathbb{P}(\max_{i,j} |W_{ij}^1| < \chi^{-\epsilon L/8} {t'}_\epsilon^{(k_x)}) &\geq \mathbb{P}(\max_{i,j} |W_{ij}^1| < Q_{1-(1-\frac{x-\epsilon}{1-\epsilon})^{\gamma_L^{2-\zeta_\epsilon}}}) + \mathcal{O}(\frac{1}{\sqrt{LN^2}}) \\ &\geq 1 - N^2(\frac{x-\epsilon}{1-\epsilon})^{\gamma_L^{2-\zeta_\epsilon}} + \mathcal{O}(\frac{1}{\sqrt{LN^2}}).\end{aligned}$$

Using the concentration inequalities on the gradient above, we have that

$$\mathbb{P}(B) \geq (1 - A\ \chi^{\epsilon L/4})^{N^2}(1 - \lambda\ \chi^{\epsilon L/2})^{LN^2}$$

so it is straightforward that there exists a constant $\eta > 0$ independent of ϵ such that

$$\mathbb{P}(B) \geq 1 - \eta L N^2 \chi^{\epsilon L/4}.$$

Therefore, we obtain

$$\mathbb{P}(s_{cr} \geq x) \leq N^2(\frac{x-\epsilon}{1-\epsilon})^{\gamma_L^{2-\zeta_\epsilon}} + \eta L N^2 \chi^{\epsilon L/4} + \mathcal{O}(\frac{1}{\sqrt{LN^2}}).$$

By integration of the previous inequality, we obtain

$$\mathbb{E}[s_{cr}] \leq \epsilon + \frac{1}{L} + \frac{N^2}{1+\gamma_L^{2-\zeta_\epsilon}} + \eta L N^2 \chi^{\epsilon L/4} + \mathcal{O}(\frac{1}{\sqrt{LN^2}}).$$

Now let $\kappa = \frac{|\log(\chi)|}{8}$. We choose $\epsilon = \frac{\log(\kappa L N^2)}{\kappa L}$. By the definition of x_{ζ_ϵ}, we have that

$$\gamma_L Q_{x_{\zeta_\epsilon,\gamma_L}} = Q_{1-(1-x_{\zeta_\epsilon,\gamma_L})^{\gamma_L^{2-\zeta_\epsilon}}}.$$

Using the asymptotic equivalent of the right hand side as $L \to \infty$, we have that $Q_{1-(1-x_{\zeta_\epsilon,\gamma_L})^{\gamma_L^{2-\zeta_\epsilon}}} \sim \sqrt{-2\log((1-x_{\zeta_\epsilon,\gamma_L})^{\gamma_L^{2-\zeta_\epsilon}})} = \gamma_L^{1-\zeta_\epsilon/2}\sqrt{-2\log(1-x_{\zeta_\epsilon,\gamma_L})}$. Therefore, we obtain

$$Q_{1-(1-x_{\zeta_\epsilon,\gamma_L})^{\gamma_L^{2-\zeta_\epsilon}}} \sim \gamma_L^{1-\zeta_\epsilon/2}\sqrt{\frac{2\log(\kappa L N^2)}{\kappa L}}.$$

For the left hand side, we have $\gamma_L Q_{x_{\zeta_\epsilon,\gamma_L}} \sim \gamma_L F'(0)\frac{\log(\kappa L N^2)}{\kappa L}$ where $F'(0)$ is the derivative at zero of the cdf of the Folded standard normal distribution. The results above prove that

$$\gamma_L^{-\zeta_\epsilon} \sim \beta\frac{\log(\kappa L N^2)}{\kappa L},$$

where β is a positive constant. This yields

$$\begin{aligned}\mathbb{E}[s_{cr}] &\leq \frac{\log(\kappa L N^2)}{\kappa L} + \frac{1}{L} + \frac{\mu}{\kappa L N^2 \log(\kappa L N^2)}(1+o(1)) + \eta\frac{1}{\kappa^2 L N^2} + \mathcal{O}(\frac{1}{\sqrt{LN^2}}) \\ &= \frac{1}{L}(1 + \frac{\log(\kappa L N^2)}{\kappa}) + \mathcal{O}(\frac{1}{\kappa^2\sqrt{LN^2}}),\end{aligned}$$

where $\kappa = \frac{|\log(\chi)|}{8}$ and μ is a constant.

2. **Case 2 : Convolutional Neural Networks**
The proof for CNNs in similar to that of FFNN once we prove that

$$\mathbb{E}\big[\frac{\partial \mathcal{L}}{\partial W_{i,j,\beta}^l}^2\big] = A\ \chi^{L-l}$$

where A is a constant. We have that

$$\frac{\partial \mathcal{L}}{\partial W^l_{i,j,\beta}} = \sum_\alpha \frac{\partial \mathcal{L}}{\partial y^l_{i,\alpha}} \phi(y^{l-1}_{j,\alpha+\beta})$$

and

$$\frac{\partial \mathcal{L}}{\partial y^l_{i,\alpha}} = \sum_{j=1}^{n} \sum_{\beta \in ker} \frac{\partial \mathcal{L}}{\partial y^{l+1}_{j,\alpha-\beta}} W^{l+1}_{i,j,\beta} \phi'(y^l_{i,\alpha}).$$

Using the hypothesis of independence of forward and backward weights and averaging over the number of channels (using CLT) we have that

$$\mathbb{E}[\frac{\partial \mathcal{L}}{\partial y^l_{i,\alpha}}^2] = \frac{\sigma_w^2 \mathbb{E}[\phi'(\sqrt{q}Z)^2]}{2k+1} \sum_{\beta \in ker} \mathbb{E}[\frac{\partial \mathcal{L}}{\partial y^{l+1}_{i,\alpha-\beta}}^2].$$

Summing over α and using the periodic boundary condition, this yields

$$\sum_\alpha \mathbb{E}[\frac{\partial \mathcal{L}}{\partial y^l_{i,\alpha}}^2] = \chi \sum_\alpha \mathbb{E}[\frac{\partial \mathcal{L}}{\partial y^{l+1}_{i,\alpha}}^2].$$

Here also, on the Ordered phase, the variance q^l, the correlation c^l, and the correlation of the gradients $\tilde{c}^l$ converge exponentially to their limiting values $q, 1$ and 0 respectively. We use the following approximations

- $\forall x \neq x', c^l_{xx'} \approx 1$,
- $\forall x, q^l_{xx} \approx q$.

Using these approximations, we have

$$\mathbb{E}\Big[\frac{\partial \mathcal{L}}{\partial W^l_{i,j,\beta}}^2\Big] = \mathbb{E}[\phi(\sqrt{q}Z)^2] \tilde{q}^l_x,$$

where $\tilde{q}^l_x = \sum_\alpha \mathbb{E}[\frac{\partial \mathcal{L}}{\partial y^l_{i,\alpha}(x)}^2]$ for an input x. The choice of x is not important in our approximation.

From the analysis above, we have

$$\tilde{q}^l_x = \tilde{q}^L_x \ \chi^{L-l}$$

we conclude that

$$\mathbb{E}\Big[\frac{\partial \mathcal{L}}{\partial W^l_{i,j,\beta}}^2\Big] = A\ \chi^{L-l}$$

where $A = \mathbb{E}[\phi(\sqrt{q}Z)^2]\tilde{q}_x^L$.

□

C Proofs for Section 3 : Training sparse networks and the rescaling trick

Proposition 7 (Rescaling Trick)**.** *Consider a neural network of the form 10 or 11 (FFNN or CNN) initialized on the EOC. Then, after pruning, the sparse network is not initialized on the EOC. However, the rescaled sparse network*

$$y^l(x) = \mathcal{F}(\rho^l \circ \delta^l \circ W^l, y^{l-1}(x)) + B^l, \quad \text{for } l \geq 1, \tag{13}$$

where

- $\rho^l_{ij} = \frac{1}{\sqrt{\mathbb{E}[N_{l-1}(W^l_{i1})^2\delta^l_{i1}]}}$ *for FFNN of the form 10,*
- $\rho^l_{i,j,\beta} = \frac{1}{\sqrt{\mathbb{E}[n_{l-1}(W^l_{i,1,\beta})^2\delta^l_{i,1,\beta}]}}$ *for CNN of the form 11,*

is initialized on the EOC.

Proof. For two inputs x, x', the forward propagation of the covariance is given by

$$\begin{aligned}\hat{q}^l(x,x') &= \mathbb{E}[y^l_i(x)y^l_i(x')] \\ &= \mathbb{E}[\sum_{j,k}^{N_{l-1}} W^l_{ij}W^l_{ik}\delta^l_{ij}\delta^l_{ik}\phi(\hat{y}^{l-1}_j(x))\phi(\hat{y}^{l-1}_j(x'))] + \sigma_b^2.\end{aligned}$$

We have that

$$\begin{aligned}\frac{\partial \mathcal{L}}{\partial W^l_{ij}} &= \frac{1}{|\mathcal{D}|}\sum_{x\in\mathcal{D}} \frac{\partial \mathcal{L}}{\partial y^l_i(x)}\frac{\partial y^l_i(x)}{\partial W^l_{ij}} \\ &= \frac{1}{|\mathcal{D}|}\sum_{x\in\mathcal{D}} \frac{\partial \mathcal{L}}{\partial y^l_i(x)}\phi(y^{l-1}_j(x)).\end{aligned}$$

Under the assumption that the weights used for forward propagation are independent from the weights used for back-propagation, we have that W^l_{ij} and $\frac{\partial \mathcal{L}}{\partial y^l_i(x)}$ are independent

for all $x \in \mathcal{D}$. We also have that W_{ij}^l and $\phi(y_j^{l-1}(x))$ are independent for all $x \in \mathcal{D}$, therefore, W_{ij}^l and $\frac{\partial \mathcal{L}}{\partial W_{ij}^l}$ are independent for all l, i, j. This yields

$$\hat{q}^l(x, x') = \sigma_w^2 \alpha_l \mathbb{E}[\phi(\hat{y}_1^{l-1}(x))\phi(\hat{y}_1^{l-1}(x'))] + \sigma_b^2,$$

where $\alpha_l = \mathbb{E}[N_{l-1}(W_{11}^l)^2 \delta_{11}^l]$ (the choice of i, j does not matter because they are iid). Unless we do not prune any weights from the l^{th} layer, we have that $\alpha_l < 1$.
These dynamics are the same as a FFNN with the variance of the weights given by $\hat{\sigma}_w^2 = \sigma_w^2 \alpha_l$. Since the EOC equation is given by $\sigma_w^2 \mathbb{E}[\phi'(\sqrt{q}Z)^2] = 1$, with the new variance, it is clear that $\hat{\sigma}_w^2 \mathbb{E}[\phi'(\sqrt{\hat{q}}Z)^2] \neq 1$ in general. Hence, the network is no longer on the EOC and this could be problematic for training.
With the rescaling, this becomes

$$\begin{aligned}\hat{q}^l(x, x') &= \sigma_w^2 \rho_l^2 \alpha_l \mathbb{E}[\phi(\tilde{y}_1^{l-1}(x))\phi(\tilde{y}_1^{l-1}(x'))] + \sigma_b^2 \\ &= \sigma_w^2 \mathbb{E}[\phi(\tilde{y}_1^{l-1}(x))\phi(\tilde{y}_1^{l-1}(x'))] + \sigma_b^2.\end{aligned}$$

Therefore, the new variance after re-scaling is $\tilde{\sigma}_w^2 = \sigma_w^2$, and the limiting variance $\tilde{q} = q$ remains also unchanged since the dynamics are the same. Therefore $\tilde{\sigma}_w^2 \mathbb{E}[\phi'(\sqrt{\tilde{q}}Z)^2] = \sigma_w^2 \mathbb{E}[\phi'(\sqrt{q}Z)^2] = 1$. Thus, the re-scaled network is initialized on the EOC. The proof is similar for CNNs.

□

D Proof for section 4 : Pruning Residual Networks

Theorem 4 (Resnet pruning)**.** *Consider a Resnet with either Fully Connected or Convolutional layers and ReLU activation function. Then for all $\sigma_w > 0$, the Resnet is well-conditioned. Moreover, for all $l \in \{1, ..., L\}, m^l = \Theta((1 + \frac{\sigma_w^2}{2})^L)$.*

Proof. Let us start with the case of a Resnet with Fully Connected layers. we have that

$$\begin{aligned}\frac{\partial \mathcal{L}}{\partial W_{ij}^l} &= \frac{1}{|\mathcal{D}|} \sum_{x \in \mathcal{D}} \frac{\partial \mathcal{L}}{\partial y_i^l(x)} \frac{\partial y_i^l(x)}{\partial W_{ij}^l} \\ &= \frac{1}{|\mathcal{D}|} \sum_{x \in \mathcal{D}} \frac{\partial \mathcal{L}}{\partial y_i^l(x)} \phi(y_j^{l-1}(x))\end{aligned}$$

and the backpropagation of the gradient is given by the set of equations

$$\frac{\partial \mathcal{L}}{\partial y_i^l} = \frac{\partial \mathcal{L}}{\partial y_i^{l+1}} + \phi'(y_i^l) \sum_{j=1}^{N_{l+1}} \frac{\partial \mathcal{L}}{\partial y_j^{l+1}} W_{ji}^{l+1}.$$

Let $q_x^l = \mathbb{E}[y_i^l(x)^2]$ and $\tilde{q}_{x,x'}^l = \mathbb{E}[\frac{\partial \mathcal{L}}{\partial y_i^l(x)} \frac{\partial \mathcal{L}}{\partial y_i^l(x')}]$ for some inputs x, x'. We have that

$$q_x^l = \mathbb{E}[y_i^{l-1}(x)^2] + \sigma_w^2 \mathbb{E}[\phi(y_1^{l-1})^2] = (1 + \frac{\sigma_w^2}{2}) q_x^{l-1},$$

and

$$\tilde{q}_{x,x'}^l = (1 + \sigma_w^2 \mathbb{E}[\phi'(y_i^l(x))\phi'(y_i^l(x'))]) \tilde{q}_{x,x'}^{l+1}.$$

We also have

$$\mathbb{E}[\frac{\partial \mathcal{L}}{\partial W_{ij}^l}^2] = \frac{1}{|\mathcal{D}|^2} \sum_{x,x'} t_{x,x'}^l,$$

where $t_{x,x'}^l = \tilde{q}_{x,x'}^l \sqrt{q_x^l q_{x'}^l} f(c^{l-1}(x,x'))$ and f is defined in the preliminary results. Let $k \in \{1, 2, ..., L\}$ be fixed. We compare the terms $t_{x,x'}^l$ for $l = k$ and $l = L$. The ratio between the two terms is given by (after simplification)

$$\frac{t_{x,x'}^k}{t_{x,x'}^L} = \frac{\prod_{l=k}^{L-1} (1 + \frac{\sigma_w^2}{2} f'(c^l(x,x')))}{(1 + \frac{\sigma_w^2}{2})^{L-k}} \frac{f(c^{k-1}(x,x'))}{f(c^{L-1}(x,x'))}.$$

Since $f'(c^l(x,x)) = 1$, $f'(c^l(x,x')) = 1 - l^{-1} + o(l^{-1})$ and $f(c^l(x,x)) = 1 - sl^{-2} + o(l^{-2})$, there exist two constants $A, B > 0$ such that $A < \frac{\prod_{l=k}^{L-1} (1 + \frac{\sigma_w^2}{2} f'(c^l(x,x')))}{(1 + \frac{\sigma_w^2}{2})^{L-k}} < B$ for all L and $k \in \{1, 2, ..., L\}$. This yields

$$A \leq \frac{\mathbb{E}[\frac{\partial \mathcal{L}}{\partial W_{ij}^l}^2]}{\mathbb{E}[\frac{\partial \mathcal{L}}{\partial W_{ij}^L}^2]} \leq B,$$

which concludes the proof.

For Resnet with convolutional layers, we have

$$\frac{\partial \mathcal{L}}{\partial W_{i,j,\beta}^l} = \frac{1}{|\mathcal{D}|} \sum_{x \in \mathcal{D}} \sum_{\alpha} \frac{\partial \mathcal{L}}{\partial y_{i,\alpha}^l(x)} \phi(y_{j,\alpha+\beta}^{l-1}(x))$$

and

$$\frac{\partial \mathcal{L}}{\partial y_{i,\alpha}^l} = \frac{\partial \mathcal{L}}{\partial y_{i,\alpha}^{l+1}} + \sum_{j=1}^{n} \sum_{\beta \in ker} \frac{\partial \mathcal{L}}{\partial y_{j,\alpha-\beta}^{l+1}} W_{i,j,\beta}^{l+1} \phi'(y_{i,\alpha}^l).$$

Let $\tilde{q}^l_{\alpha,\alpha'}(x,x') = \mathbb{E}[\frac{\partial \mathcal{L}}{\partial y^l_{i,\alpha}(x)} \frac{\partial \mathcal{L}}{\partial y^l_{i,\alpha'}(x')}]$.

Using the hypothesis of independence of forward and backward weights and averaging over the number of channels (using CLT), we have that

$$\tilde{q}^l_{\alpha,\alpha'}(x,x') = \tilde{q}^{l+1}_{\alpha,\alpha'}(x,x') + \frac{\sigma_w^2 f'(c^l_{\alpha,\alpha'}(x,x'))}{2(2k+1)} \sum_\beta \tilde{q}^{l+1}_{\alpha+\beta,\alpha'+\beta}(x,x').$$

Let $K_l = ((\tilde{q}^l_{\alpha,\alpha+\beta}(x,x'))_{\alpha\in[|0,N-1|]})_{\beta\in[|0,N-1]}$ is a vector in $\mathbb{R}^{N^2}$. Writing this previous equation in matrix form, we have

$$K_l = (I + \frac{\sigma_w^2 f'(c^l_{\alpha,\alpha'}(x,x'))}{2(2k+1)} U) K_{l+1}$$

and

$$\mathbb{E}[\frac{\partial \mathcal{L}}{\partial W^l_{i,j,\beta}}^2] = \frac{1}{|\mathcal{D}|^2} \sum_{x,x'\in\mathcal{D}} \sum_{\alpha,\alpha'} t^l_{\alpha,\alpha'}(x,x'),$$

where $t^l_{\alpha,\alpha'}(x,x') = \tilde{q}^l_{\alpha,\alpha'}(x,x')\sqrt{q^l_{\alpha+\beta}(x) q^l_{\alpha'+\beta}(x')} f(c^{l-1}_{\alpha+\beta,\alpha'+\beta}(x,x'))$. Since $f'(c^l_{\alpha,\alpha'}(x,x')) \to 1$, then by fixing l and letting L goes to infinity, we have that

$$K_l \sim_{L\to\infty} (1+\frac{\sigma_w^2}{2})^{L-l} e_1 e_1^T K_L$$

and, from Lemma 4, we know that

$$\sqrt{q^l_{\alpha+\beta}(x) q^l_{\alpha'+\beta}(x')} = (1+\frac{\sigma_w^2}{2})^{l-1} \sqrt{q_{0,x} q_{0,x'}}.$$

Therefore, for a fixed $k < L$, we have $t^k_{\alpha,\alpha'}(x,x') \sim (1+\frac{\sigma_w^2}{2})^{L-1} f(c^{k-1}_{\alpha+\beta,\alpha'+\beta}(x,x'))(e_1^T K_L) = \Theta(t^L_{\alpha,\alpha'}(x,x'))$ which concludes the proof.

□

Proposition 8 (Stable Resnet)**.** *Consider the following Resnet parameterization*

$$y^l(x) = y^{l-1}(x) + \frac{1}{\sqrt{L}} \mathcal{F}(W^l, y^{l-1}), \quad \text{for } l \geq 2, \tag{14}$$

then the network is well-conditioned for all choices of $\sigma_w > 0$. Moreover, for all $l \in \{1,...,L\}$ we have $m^l = \Theta(L^{-1})$.

Proof. The proof is similar to that of theorem 4 with minor differences. Let us start with the case of a Resnet with Fully Connected layers, we have

$$\frac{\partial \mathcal{L}}{\partial W_{ij}^l} = \frac{1}{|\mathcal{D}|\sqrt{L}} \sum_{x \in \mathcal{D}} \frac{\partial \mathcal{L}}{\partial y_i^l(x)} \frac{\partial y_i^l(x)}{\partial W_{ij}^l}$$
$$= \frac{1}{|\mathcal{D}|\sqrt{L}} \sum_{x \in \mathcal{D}} \frac{\partial \mathcal{L}}{\partial y_i^l(x)} \phi(y_j^{l-1}(x))$$

and the backpropagation of the gradient is given by

$$\frac{\partial \mathcal{L}}{\partial y_i^l} = \frac{\partial \mathcal{L}}{\partial y_i^{l+1}} + \frac{1}{\sqrt{L}} \phi'(y_i^l) \sum_{j=1}^{N_{l+1}} \frac{\partial \mathcal{L}}{\partial y_j^{l+1}} W_{ji}^{l+1}.$$

Let $q_x^l = \mathbb{E}[y_i^l(x)^2]$ and $\tilde{q}_{x,x'}^l = \mathbb{E}[\frac{\partial \mathcal{L}}{\partial y_i^l(x)} \frac{\partial \mathcal{L}}{\partial y_i^l(x')}]$ for some inputs x, x'. We have

$$q_x^l = \mathbb{E}[y_i^{l-1}(x)^2] + \frac{\sigma_w^2}{L} \mathbb{E}[\phi(y_1^{l-1})^2] = (1 + \frac{\sigma_w^2}{2L}) q_x^{l-1}$$

and

$$\tilde{q}_{x,x'}^l = (1 + \frac{\sigma_w^2}{L} \mathbb{E}[\phi'(y_i^l(x)) \phi'(y_i^l(x'))]) \tilde{q}_{x,x'}^{l+1},$$

We also have

$$\mathbb{E}[\frac{\partial \mathcal{L}}{\partial W_{ij}^l}^2] = \frac{1}{L|\mathcal{D}|^2} \sum_{x,x'} t_{x,x'}^l,$$

where $t_{x,x'}^l = \tilde{q}_{x,x'}^l \sqrt{q_x^l q_{x'}^l} f(c^{l-1}(x, x'))$ and f is defined in the preliminary results. Let $k \in \{1, 2, ..., L\}$ be fixed. We compare the terms $t_{x,x'}^l$ for $l = k$ and $l = L$. The ratio between the two terms is given by (after simplification)

$$\frac{t_{x,x'}^k}{t_{x,x'}^L} = \frac{\prod_{l=k}^{L-1} (1 + \frac{\sigma_w^2}{2L} f'(c^l(x, x')))}{(1 + \frac{\sigma_w^2}{2L})^{L-k}} \frac{f(c^{k-1}(x, x'))}{f(c^{L-1}(x, x'))}.$$

Since $f'(c^l(x, x)) = 1$, $f'(c^l(x, x')) = 1 - l^{-1} + o(l^{-1})$ and $f(c^l(x, x)) = 1 - sl^{-2} + o(l^{-2})$, there exists two constants $A, B > 0$ such that $A < \frac{\prod_{l=k}^{L-1} (1 + \frac{\sigma_w^2}{2} f'(c^l(x, x')))}{(1 + \frac{\sigma_w^2}{2})^{L-k}} < B$ for all L and $k \in \{1, 2, ..., L\}$. This yields

$$A \leq \frac{\mathbb{E}[\frac{\partial \mathcal{L}}{\partial W_{ij}^l}^2]}{\mathbb{E}[\frac{\partial \mathcal{L}}{\partial W_{ij}^L}^2]} \leq B.$$

Moreover, since $(1+\frac{\sigma_w^2}{2})^L \to e^{\sigma_w^2/2}$, then $m^l = \Theta(1)$ for all $l \in \{1, ..., L\}$, which concludes the proof.

For Resnet with convolutional layers, the proof is similar. With the scaling, we have

$$\frac{\partial \mathcal{L}}{\partial W^l_{i,j,\beta}} = \frac{1}{\sqrt{L}|\mathcal{D}|} \sum_{x \in \mathcal{D}} \sum_{\alpha} \frac{\partial \mathcal{L}}{\partial y^l_{i,\alpha}(x)} \phi(y^{l-1}_{j,\alpha+\beta}(x))$$

and

$$\frac{\partial \mathcal{L}}{\partial y^l_{i,\alpha}} = \frac{\partial \mathcal{L}}{\partial y^{l+1}_{i,\alpha}} + \frac{1}{\sqrt{L}} \sum_{j=1}^{n} \sum_{\beta \in ker} \frac{\partial \mathcal{L}}{\partial y^{l+1}_{j,\alpha-\beta}} W^{l+1}_{i,j,\beta} \phi'(y^l_{i,\alpha}).$$

Let $\tilde{q}^l_{\alpha,\alpha'}(x,x') = \mathbb{E}[\frac{\partial \mathcal{L}}{\partial y^l_{i,\alpha}(x)} \frac{\partial \mathcal{L}}{\partial y^l_{i,\alpha'}(x')}]$. Using the hypothesis of independence of forward and backward weights and averaging over the number of channels (using CLT) we have that

$$\tilde{q}^l_{\alpha,\alpha'}(x,x') = \tilde{q}^{l+1}_{\alpha,\alpha'}(x,x') + \frac{\sigma_w^2 f'(c^l_{\alpha,\alpha'}(x,x'))}{2(2k+1)L} \sum_{\beta} \tilde{q}^{l+1}_{\alpha+\beta,\alpha'+\beta}(x,x').$$

Let $K_l = ((\tilde{q}^l_{\alpha,\alpha+\beta}(x,x'))_{\alpha \in [|0,N-1|]})_{\beta \in [|0,N-1]}$ is a vector in $\mathbb{R}^{N^2}$. Writing this previous equation in matrix form, we have

$$K_l = (I + \frac{\sigma_w^2 f'(c^l_{\alpha,\alpha'}(x,x'))}{2(2k+1)L} U) K_{l+1},$$

and

$$\mathbb{E}[\frac{\partial \mathcal{L}}{\partial W^l_{i,j,\beta}}^2] = \frac{1}{L|\mathcal{D}|^2} \sum_{x,x' \in \mathcal{D}} \sum_{\alpha,\alpha'} t^l_{\alpha,\alpha'}(x,x'),$$

where $t^l_{\alpha,\alpha'}(x,x') = \tilde{q}^l_{\alpha,\alpha'}(x,x') \sqrt{q^l_{\alpha+\beta}(x) q^l_{\alpha'+\beta}(x')} f(c^{l-1}_{\alpha+\beta,\alpha'+\beta}(x,x'))$. Since $f'(c^l_{\alpha,\alpha'}(x,x')) \to 1$, then by fixing l and letting L goes to infinity, we have that

$$K_l \sim_{L \to \infty} (1 + \frac{\sigma_w^2}{2L})^{L-l} e_1 e_1^T K_L$$

and we know from appendix lemma 4 that

$$\sqrt{q^l_{\alpha+\beta}(x) q^l_{\alpha'+\beta}(x')} = (1 + \frac{\sigma_w^2}{2L})^{l-1} \sqrt{q_{0,x} q_{0,x'}}.$$

Therefore, for a fixed $k < L$, we have $t^k_{\alpha,\alpha'}(x,x') \sim (1+\frac{\sigma_w^2}{2L})^{L-1} f(c^{k-1}_{\alpha+\beta,\alpha'+\beta}(x,x'))(e_1^T K_L) = \Theta(t^L_{\alpha,\alpha'}(x,x'))$ which proves that the stable resnet is well conditioned. Moreover, since $(1+\frac{\sigma_w^2}{2L})^{L-1} \to e^{\sigma_w^2/2}$, then $m^l = \Theta(L^{-1})$ for all l.

□

Proposition 9 (Resnet live on the EOC even after pruning)**.** *Let x, x' be two inputs. The following statments hold*

1. *For Resnet with Fully Connected layers, let $\hat{c}^l(x, x')$ be the correlation between $\hat{y}^l_i(x)$ and $\hat{y}^l_i(x')$ after pruning the network. Then we have*

$$1 - \hat{c}^l(x, x') \sim \frac{\kappa}{l^2},$$

where $\kappa > 0$ is a constant.

2. *For Resnet with Convolutional layers, let $\hat{c}^l(x, x') = \frac{\sum_{\alpha,\alpha'} \mathbb{E}[y^l_{1,\alpha}(x) y^l_{1,\alpha'}(x')]}{\sum_{\alpha,\alpha'} \sqrt{q^l_\alpha(x)}\sqrt{q^l_{\alpha'}(x')}}$ be an 'average' correlation after pruning the network. Then we have*

$$1 - \hat{c}^l(x, x') \gtrsim l^{-2}.$$

Proof. • Let x and x' be two inputs. The covariance of $\hat{y}^l_i(x)$ and $\hat{y}^l_i(x')$ is given by

$$\hat{q}^l(x, x') = \hat{q}^{l-1}(x, x') + \alpha \mathbb{E}_{(Z_1,Z_2)\sim\mathcal{N}(0,Q^{l-1})}[\phi(Z_1)\phi(Z_2)]$$

where $Q^{l-1} = \begin{bmatrix} \hat{q}^{l-1}(x) & \hat{q}^{l-1}(x, x') \\ \hat{q}^{l-1}(x, x') & \hat{q}^{l-1}(x') \end{bmatrix}$ and $\alpha = \mathbb{E}[N_{l-1} {W^l}_{11}^2 \delta^l_{11}]$.

Consequently, we have that $\hat{q}^l(x) = (1 + \frac{\alpha}{2})\hat{q}^{l-1}(x)$. Therefore, we have

$$\hat{c}^l(x, x') = \frac{1}{1+\lambda}\hat{c}^{l-1}(x, x') + \frac{\lambda}{1+\lambda} f(\hat{c}^{l-1}(x, x')),$$

where $\lambda = \frac{\alpha}{2}$ and $f(x) = 2\mathbb{E}[\phi(Z_1)\phi(xZ_1 + \sqrt{1-x^2}Z_2)]$ and Z_1 and Z_2 are iid standard normal variables.

Using preliminary results, We have that $\hat{c}^l(x, x') \to 1$. Let $\zeta_l = 1 - \hat{c}^l(x, x')$. Using the fact that $f(x) \underset{x\to 1^-}{=} x + \beta(1-x)^{3/2} + O((1-x)^{5/2})$, we have that

$$\zeta_l = \zeta_{l-1} - \eta\zeta_{l-1}^{3/2} + O(\zeta_{l-1}^{5/2}),$$

where $\eta = \frac{\lambda\beta}{1+\lambda}$. Now using the asymptotic development of $\zeta_l^{-1/2}$ given by

$$\zeta_l^{-1/2} = \zeta_{l-1}^{-1/2} + \frac{\eta}{2} + O(\zeta_{l-1}),$$

this yields $\zeta_l^{-1/2} \underset{l\to\infty}{\sim} \frac{\eta}{2} l$. We conclude that $1 - \hat{c}^l_{ab} \sim \frac{4}{\eta^2 l^2}$.

- For some input x, recall the forward propagation of a pruned 1D convolutional neural network

$$y^l_{i,\alpha}(x) = y^{l-1}_{i,\alpha}(x) + \sum_{j=1}^{c} \sum_{\beta \in ker} \delta^l_{i,j.\beta} W^l_{i,j,\beta} \phi(y^{l-1}_{j,\alpha+\beta}(x)) + b^l_i.$$

Unlike FFNN, neurons in the same channel are correlated since we use the same filters for all of them. Let x, x' be two inputs and α, α' two nodes in the same channel i. Using Central Limit Theorem in the limit of large c (number of channels), we have

$$\mathbb{E}[y^l_{i,\alpha}(x) y^l_{i,\alpha'}(x')] = \mathbb{E}[y^{l-1}_{i,\alpha}(x) y^{l-1}_{i,\alpha'}(x')] + \frac{1}{2k+1} \sum_{\beta \in ker} \alpha_\beta \mathbb{E}[\phi(y^{l-1}_{1,\alpha+\beta}(x)) \phi(y^{l-1}_{1,\alpha'+\beta}(x'))].$$

where $\alpha_\beta = \mathbb{E}[\delta^l_{i,1.\beta} {W^l}^2_{i,1,\beta} n_{l-1}]$.
Let $q^l_\alpha(x) = \mathbb{E}[y^l_{1,\alpha}(x)^2]$. The choice of the channel is not important since for a given α, neurons $(y^l_{i,\alpha}(x))_{i \in [c]}$ are iid. Using the previous formula, we have that

$$\begin{aligned} q^l_\alpha(x) &= q^{l-1}_\alpha(x) + \frac{1}{2k+1} \sum_{\beta \in ker} \alpha_\beta \mathbb{E}[\phi(y^{l-1}_{1,\alpha+\beta}(x))^2] \\ &= q^{l-1}_\alpha(x) + \frac{1}{2k+1} \sum_{\beta \in ker} \alpha_\beta \frac{q^{l-1}_{\alpha+\beta}(x)}{2}. \end{aligned}$$

Therefore, letting $q^l(x) = \frac{1}{N} \sum_{\alpha \in [N]} q^l_\alpha(x)$ and $\sigma = \frac{\sum_\beta \alpha_\beta}{2k+1}$, we have that

$$\begin{aligned} q^l(x) &= q^{l-1}(x) + \frac{1}{2k+1} \sum_{\beta \in ker} \alpha_\beta \sum_{\alpha \in [n]} \frac{q^{l-1}_{\alpha+\beta}(x)}{2} \\ &= (1 + \frac{\sigma}{2}) q^{l-1}(x) = (1 + \frac{\sigma}{2})^{l-1} q^1(x), \end{aligned}$$

where we have used the periodicity $q^{l-1}_\alpha = q^{l-1}_{\alpha-N} = q^{l-1}_{\alpha+N}$. Moreover, we have that $\min_\alpha q^l_\alpha(x) \geq (1 + \frac{\sigma}{2}) \min_\alpha q^{l-1}_\alpha(x) \geq (1 + \frac{\sigma}{2})^{l-1} \min_\alpha q^1_\alpha(x)$.

In the next Lemma, we study the asymptotic behaviour of the variance q^l_α. We show that as $l \to \infty$, a phenomenon of self averaging yields to the fact that q^l_α becomes independent of α.

Appendix Lemma 4. *There exists $\beta > 0$ such that for all $x \in \mathbb{R}^d$ and α,*

$$q^l_\alpha(x) = (1 + \frac{\sigma_w^2}{2})^l q_{0,x} + \mathcal{O}((1 + \frac{\sigma_w^2}{2})^l e^{-\beta l})),$$

where $q_{0,x}$ is a constant that depends on x.

Proof. Recall that

$$q_\alpha^l(x) = q_\alpha^{l-1}(x) + \frac{1}{2k+1} \sum_{\beta \in ker} \alpha_\beta \frac{q_{\alpha+\beta}^{l-1}(x)}{2},$$

we write this in a matrix form

$$A_l = U A_{l-1}$$

where $A_l = (q_\alpha^l(x))_\alpha$ is a vector in $\mathbb{R}^N$ and U is the is the convolution matrix. As an example, for $k = 1$, U given by

$$U = \begin{bmatrix} 1+\alpha_0 & \alpha_1 & 0 & ... & 0 & \alpha_{-1} \\ \alpha_{-1} & 1+\alpha_0 & \alpha_1 & 0 & \ddots & 0 \\ 0 & \alpha_{-1} & 1+\alpha_0 & \alpha_1 & \ddots & 0 \\ 0 & 0 & \alpha_{-1} & 1+\alpha_0 & \ddots & 0 \\ & \ddots & \ddots & \ddots & \ddots & \\ \alpha_1 & 0 & \dots & 0 & \alpha_{-1} & 1+\alpha_0 \end{bmatrix}$$

where $\delta = \frac{\sigma_w^2}{2(2k+1)}$. U is a circulant symmetric matrix with eigenvalues $\lambda_1 > \lambda_2 \geq \lambda_3 ... \geq \lambda_N$. The largest eigenvalue of U is given by $\lambda_1 = 1 + \sum_\beta \alpha_\beta$ and its equivalent eigenspace is generated by the vector $e_1 = \frac{1}{\sqrt{N}}(1, 1, ..., 1) \in \mathbb{R}^N$. This yields

$$(1 + \frac{\sigma}{2})^{-l} U^l = e_1 e_1^T + O(e^{-\beta l}),$$

where $\beta = \log(\frac{\lambda_1}{\lambda_2})$
Using this, we have that

$$\lambda_1^{-l} A_l = (\lambda_1^{-l} U^l) A_0 = e_1 e_1^T A_0 + O(e^{-\beta l})$$

this concludes the proof. □

The convolutional structure makes it hard to analyse the correlation between the values of a neurons for two different inputs (dependency). In Xiao et al. [2018], authors studied the correlation between the values of two neurons in the same channel for the same input. Although this could capture the propagation of the input structure (how different pixels propagate together) inside the network, it does not provide any information on how different structures from different inputs propagate. To resolve this situation, we study the 'average' correlation per channel defined as

$$c^l(x,x') = \frac{\sum_{\alpha,\alpha'} \mathbb{E}[y^l_{1,\alpha}(x) y^l_{1,\alpha'}(x')]}{\sum_{\alpha,\alpha'} \sqrt{q^l_\alpha(x)}\sqrt{q^l_{\alpha'}(x')}}.$$

We also define $\breve{c}^l(x,x')$ by

$$\breve{c}^l(x,x') = \frac{\frac{1}{N^2}\sum_{\alpha,\alpha'} \mathbb{E}[y^l_{1,\alpha}(x) y^l_{1,\alpha'}(x')]}{\sqrt{\frac{1}{N}\sum_\alpha q^l_\alpha(x)}\sqrt{\frac{1}{N}\sum_\alpha q^l_\alpha(x')}}.$$

Using the concavity of the square root function, we have that

$$\begin{aligned}
\sqrt{\frac{1}{N}\sum_\alpha q^l_\alpha(x)}\sqrt{\frac{1}{N}\sum_\alpha q^l_\alpha(x')} &= \sqrt{\frac{1}{N^2}\sum_{\alpha,\alpha'} q^l_\alpha(x) q^l_\alpha(x')} \\
&\geq \frac{1}{N^2}\sum_{\alpha,\alpha'} \sqrt{q^l_\alpha(x)}\sqrt{q^l_\alpha(x')} \\
&\geq \frac{1}{N^2}\sum_{\alpha,\alpha'} |\mathbb{E}[y^l_{1,\alpha}(x) y^l_{1,\alpha'}(x')]|.
\end{aligned}$$

This yields $\breve{c}^l(x,x') \leq c^l(x,x') \leq 1$. Using Lemma 4, there exists $\beta > 0$ such that

$$c^l(x,x') = \breve{c}^l(x,x')(1 + \mathcal{O}(e^{-\beta l})). \tag{15}$$

This results shows that studying the limiting behaviour of $c^l(x,x')$ is equivalent to that of $\breve{c}^l(x,x')$ up to an exponentially small factor. We study hereafter the behaviour of $\breve{c}^l(x,x')$ and use this result to conclude.
Recall that

$$\mathbb{E}[y^l_{i,\alpha}(x) y^l_{i,\alpha'}(x')] = \mathbb{E}[y^{l-1}_{i,\alpha}(x) y^{l-1}_{i,\alpha'}(x')] + \frac{1}{2k+1}\sum_{\beta \in ker} \alpha_\beta \mathbb{E}[\phi(y^{l-1}_{1,\alpha+\beta}(x))\phi(y^{l-1}_{1,\alpha'+\beta}(x'))].$$

Therefore,

$$\begin{aligned}
\sum_{\alpha,\alpha'} \mathbb{E}[y^l_{1,\alpha}(x) y^l_{1,\alpha'}(x')] &= \sum_{\alpha,\alpha'} \mathbb{E}[y^{l-1}_{1,\alpha}(x) y^{l-1}_{1,\alpha'}(x')] + \frac{1}{2k+1}\sum_{\alpha,\alpha'}\sum_{\beta\in ker} \alpha_\beta \mathbb{E}[\phi(y^{l-1}_{1,\alpha+\beta}(x))\phi(y^{l-1}_{1,\alpha'+\beta}(x'))] \\
&= \sum_{\alpha,\alpha'} \mathbb{E}[y^{l-1}_{1,\alpha}(x) y^{l-1}_{1,\alpha'}(x')] + \sigma \sum_{\alpha,\alpha'} \mathbb{E}[\phi(y^{l-1}_{1,\alpha}(x))\phi(y^{l-1}_{1,\alpha'}(x'))] \\
&= \sum_{\alpha,\alpha'} \mathbb{E}[y^{l-1}_{1,\alpha}(x) y^{l-1}_{1,\alpha'}(x')] + \frac{\sigma}{2}\sum_{\alpha,\alpha'} \sqrt{q^{l-1}_\alpha(x)}\sqrt{q^{l-1}_{\alpha'}(x')} f(c^{l-1}_{\alpha,\alpha'}(x,x')),
\end{aligned}$$

where f is the correlation function of ReLU.

Let us first prove that $\breve{c}^l(x,x')$ converges to 1. Using the fact that $f(z) \geq z$ for all $z \in (0,1)$, we have that

$$\begin{aligned}\sum_{\alpha,\alpha'} \mathbb{E}[y^l_{1,\alpha}(x) y^l_{1,\alpha'}(x')] &\geq \sum_{\alpha,\alpha'} \mathbb{E}[y^{l-1}_{1,\alpha}(x) y^{l-1}_{1,\alpha'}(x')] + \frac{\sigma}{2} \sum_{\alpha,\alpha'} \sqrt{q^{l-1}_{\alpha}(x)} \sqrt{q^{l-1}_{\alpha'}(x')} c^{l-1}_{\alpha,\alpha'}(x,x') \\ &= \sum_{\alpha,\alpha'} \mathbb{E}[y^{l-1}_{1,\alpha}(x) y^{l-1}_{1,\alpha'}(x')] + \frac{\sigma}{2} \sum_{\alpha,\alpha'} \mathbb{E}[y^{l-1}_{1,\alpha}(x) y^{l-1}_{1,\alpha'}(x')] \\ &= (1 + \frac{\sigma}{2}) \mathbb{E}[y^{l-1}_{1,\alpha}(x) y^{l-1}_{1,\alpha'}(x')].\end{aligned}$$

Combining this result with the fact that $\sum_\alpha q^l_\alpha(x) = (1 + \frac{\sigma}{2}) \sum_\alpha q^{l-1}_\alpha(x)$, we have $\breve{c}^l(x,x') \geq \breve{c}^{l-1}(x,x')$. Therefore $\breve{c}^l(x,x')$ is non-decreasing and converges to a limiting point c.
Let us prove that $c = 1$. By contradiction, assume the limit $c < 1$. Using equation (15), we have that $\frac{c^l(x,x')}{\breve{c}^l(x,x')}$ converge to 1 as l goes to infinity. This yields to $c^l(x,x') \to c$ also. Therefore, there exists α_0, α'_0 and a constant $\delta < 1$ such that for all l, $c^l_{\alpha_0,\alpha'_0}(x,x') \leq \delta < 1$. Knowing that f is strongly convex and that $f'(1) = 1$, we have that $f(c^l_{\alpha_0,\alpha'_0}(x,x')) \geq c^l_{\alpha_0,\alpha'_0}(x,x') + f(\delta) - \delta$. Therefore,

$$\begin{aligned}\breve{c}^l(x,x') &\geq \breve{c}^{l-1}(x,x') + \frac{\frac{\sigma}{2}\sqrt{q^{l-1}_{\alpha_0}(x) q^{l-1}_{\alpha'_0}(x')}}{N^2 \sqrt{q^l(x)} \sqrt{q^l(x')}} (f(\delta) - \delta) \\ &\geq \breve{c}^{l-1}(x,x') + \frac{\frac{\sigma}{2}\sqrt{\min_\alpha q^1_\alpha(x) \min_{\alpha'} q^1_{\alpha'}(x')}}{N^2 \sqrt{q^1(x)} \sqrt{q^1(x')}} (f(\delta) - \delta).\end{aligned}$$

By taking the limit $l \to \infty$, we find that $c \geq c + \frac{\frac{\sigma}{2}\sqrt{\min_\alpha q^1_\alpha(x) \min_{\alpha'} q^1_{\alpha'}(x')}}{N^2 \sqrt{q^1(x)} \sqrt{q^1(x')}} (f(\delta) - \delta)$. This cannot be true since $f(\delta) > \delta$. Thus we conclude that $c = 1$.

Now we study the asymptotic convergence rate. From the preliminary results, we have that

$$f(x) \underset{x \to 1^-}{=} x + \frac{2\sqrt{2}}{3\pi}(1-x)^{3/2} + O((1-x)^{5/2}).$$

Therefore, there exists $\kappa > 0$ such that,

$$f(x) \leq x + \kappa(1-x)^{3/2}.$$

Using this result, we can upper bound $c^l(x, x')$

$$\breve{c}^l(x, x') \leq \breve{c}^{l-1}(x, x') + \kappa \sum_{\alpha,\alpha'} \frac{\frac{1}{N^2}\sqrt{q_\alpha^{l-1}(x)}\sqrt{q_{\alpha'}^{l-1}(x')}}{\sqrt{q^l(x)}\sqrt{q^l(x')}}(1 - c^l_{\alpha,\alpha'}(x, x'))^{3/2}.$$

To get a polynomial convergence rate, we should have an upper bound of the form $\breve{c}^l \leq \breve{c}^{l-1} + \zeta(1 - \breve{c}^{l-1})^{1+\epsilon}$ (see below). However, the function $x^{3/2}$ is convex, so the sum cannot be upper-bounded directly (using Jensen's inequality). We use a special form of inequalities for this purpose.

Theorem 5 (Theorem 1 in **?**)**.** *Let $x_1, x_2, ...x_n > 0$. For $s > r > 0$, we have that*

$$\Big(\sum_i x_i^s\Big)^{1/s} < \Big(\sum_i x_i^r\Big)^{1/r}.$$

Let $z^l_{\alpha,\alpha'} = \frac{\frac{1}{N^2}\sqrt{q_\alpha^{l-1}(x)}\sqrt{q_{\alpha'}^{l-1}(x')}}{\sqrt{q^l(x)}\sqrt{q^l(x')}}$, we have that

$$\sum_{\alpha,\alpha'} z^l_{\alpha,\alpha'}(1 - c^l_{\alpha,\alpha'}(x, x'))^{3/2} \leq \zeta_l \sum_{\alpha,\alpha'} [z^l_{\alpha,\alpha'}(1 - c^l_{\alpha,\alpha'}(x, x'))]^{3/2},$$

where $\zeta_l = \max_{\alpha,\alpha'} \frac{1}{z^{l\,1/2}_{\alpha,\alpha'}}$. Using the inequality (5) with $s = 3/2$ and $r = 1$, we have that

$$\begin{aligned}\sum_{\alpha,\alpha'} [z^l_{\alpha,\alpha'}(1 - c^l_{\alpha,\alpha'}(x, x'))]^{3/2} &\leq (\sum_{\alpha,\alpha'} z^l_{\alpha,\alpha'}(1 - c^l_{\alpha,\alpha'}(x, x')))^{3/2} \\ &= (\sum_{\alpha,\alpha'} z^l_{\alpha,\alpha'} - \breve{c}^l(x, x')))^{3/2}.\end{aligned}$$

Moreover, using the concavity of the square root function, we have that $\sum_{\alpha,\alpha'} z^l_{\alpha,\alpha'} \leq 1$. This yields

$$\breve{c}^l(x, x') \leq \breve{c}^{l-1}(x, x') + \zeta(1 - \breve{c}^{l-1}(x, x'))^{3/2},$$

where ζ is constant. Letting $\gamma_l = 1 - \breve{c}^l(x, x')$, we end up this time with this inequality (we had an equality in the case of FFNN)

$$\gamma_l \geq \gamma_{l-1} - \zeta\gamma_{l-1}^{3/2}$$

which leads to

$$\gamma_l^{-1/2} \leq \gamma_{l-1}^{-1/2}(1 - \zeta\gamma_{l-1}^{1/2})^{-1/2} = \gamma_{l-1}^{-1/2} + \frac{\zeta}{2} + o(1).$$

We conclude that

$$\gamma_l \gtrsim l^{-2}.$$

Using this result combined with equation 15 again, we conclude that

$$1 - c^l(x, x') \gtrsim l^{-2}.$$

□

E Proofs for Section 5 : The Lottery Ticket Hypothesis

Proposition 10 (Winning Tickets on the Edge of Chaos)**.** *Consider a neural network with layers initialized with variances $\sigma_{w,l} \in \mathbb{R}^+$ for each layer and variance $\sigma_b > 0$ for bias. We define $\sigma_{w,EOC}$ to be the value of σ_w such that $(\sigma_{w,EOC}, \sigma_b) \in EOC$. Then, for all sequences $(\sigma_{w,l})_l$ such that $\sigma_{w,l} > \sigma_{w,EOC}$ for all l, there exists a distribution of subnetworks initialized on the Edge of Chaos.*

Proof. We prove the result for FFNN. The proof for CNN is similar. Let x, x' be two inputs. For all l, let $(\delta^l)_{ij}$ be a collection of Bernoulli variables with probability p_l. The forward propagation of the covariance is given by

$$\begin{aligned}\hat{q}^l(x, x') &= \mathbb{E}[y_i^l(x) y_i^l(x')] \\ &= \mathbb{E}[\sum_{j,k}^{N_{l-1}} W_{ij}^l W_{ik}^l \delta_{ij}^l \delta_{ik}^l \phi(\hat{y}_j^{l-1}(x)) \phi(\hat{y}_j^{l-1}(x'))] + \sigma_b^2.\end{aligned}$$

This yields

$$\hat{q}^l(x, x') = \sigma_{w,l}^2 p_l \mathbb{E}[\phi(\hat{y}_1^{l-1}(x)) \phi(\hat{y}_1^{l-1}(x'))] + \sigma_b^2.$$

By choosing $p_l = \frac{\sigma_{w,EOC}^2}{\sigma_{w,l}^2}$, this becomes

$$\hat{q}^l(x, x') = \sigma_{w,EOC}^2 \mathbb{E}[\phi(\tilde{y}_1^{l-1}(x)) \phi(\tilde{y}_1^{l-1}(x'))] + \sigma_b^2.$$

Therefore, the new variance after pruning with the Bernoulli mask δ is $\tilde{\sigma}_w^2 = \sigma_{w,EOC}^2$. Thus, the subnetwork defined by δ is initialized on the EOC. The distribution of these subnetworks is directly linked to the distribution of δ. We can see this result as layer-wise pruning, i.e. pruning each layer aside. The proof is similar for CNNs.

□

F Additional theoretical results

Proposition 11 (MBP in the large width limit)**.** *Assume there exists $l_0 \in [|1, L|]$ such that $\alpha_{l_0} > \alpha_l$ (i.e. $v_{l_0} > v_l$) for all l, and let $s_0 = \frac{M_{l_0}}{\sum_l M_l}$. For some sparsity s, let $PR_{l_0}(s)$ be the event that layer l_0 is fully pruned before other layers, i.e.*

$$PR_{l_0}(s) = \{|A_{l_0}| = M_{l_0}\} \cap_{l \in [|1,L|]} \{|A_l| < M_l\},$$

and let $PR_{l_0} = \cup_{s \in (s_0, s_{\max})} PR_{l_0}(s)$ the event where there exists a sparsity s such that layer l_0 is fully pruned before other layers. Then, we have

$$\mathbb{P}(PR_{l_0}) \geq 1 - \frac{L\pi^2}{4(\gamma - 1)^2 \log(N)^2} + o\Big(\frac{1}{\log(N)^2}\Big),$$

where $\gamma = \min_{k \neq l_0} \frac{\alpha_{l_0}}{\alpha_k}$.

Proposition 11 shows that when the width is not the same for all layers, magnitude based pruning will result in one layer being fully pruned with a probability that converges to 1 as the width goes to infinity. The larger the ratio γ (ratio of widths between the largest and the second largest layers), the faster this probability goes to 1.

The intuition behind Proposition 11 comes from a result from Extreme Value Theory. Indeed, the problem of pruning one whole layer before the others is essentially a problem of maxima: we prune one whole layer l_0 before the others if and only if for all $\max_i |W_i^{l_0}| < \min_{l \neq l_0} \max_i |W_i^l|$. In **?**, the expected value of n iid standard Gaussian variables scales as $\sqrt{\log n}$ for large n.

Proof. Assume there exists $l_0 \in [|1, L|]$ such that $\alpha_{l_0} > \alpha_l$ for all l. The trick is to see that

$$PR_{l_0} = \{\forall k \neq l_0, \max_i |W_i^{l_0}| < \max_{ij} |W_i^k|\}.$$

Let us prove that

$$\mathbb{P}(PR_{l_0}) \geq \prod_{k \neq l_0} \mathbb{P}(\max_i |W_i^{l_0}| < \max_j |W_i^k|).$$

To establish this result, we use the following Rearrangement inequality from Hardy et al. [1952].

Appendix Lemma 5 (Rearrangement inequality)**.** *Let $f, g : \mathbb{R} \to \mathbb{R}^+$ be functions which are either both non-decreasing or non-increasing and let X be a random variable. Then*

$$\mathbb{E}[f(X)g(X)] \geq \mathbb{E}[f(X)]\mathbb{E}[g(X)].$$

Let $X = \max_i |W_i^{l_0}|$. We have that

$$\mathbb{P}(PR_{l_0}) = \mathbb{E}[\prod_{k \neq l_0} \mathbb{P}(X < \max_i |W_i^k||X)]$$

using the Rearrangement Inequality with functions $f_i(x) = \mathbb{P}(X < \max_i |W_i^k||X = x)$ which are all non-increasing, we have that

$$\mathbb{P}(PR_{l_0}) \geq \prod_{k \neq l_0} \mathbb{E}[\mathbb{P}(X < \max_i |W_i^k||X)] = \prod_{k \neq l_0} \mathbb{P}(\max_i |W_i^{l_0}| < \max_i |W_i^k|).$$

In order to deal with the probability $\mathbb{P}(\max_i |W_i^{l_0}| < \max_i |W_i^k|)$, we use a result from Extreme Value Theory.

Proposition 1 (Richard von Mises (1936))**.** *Let $(X_i)_{1 \leq i \leq n}$ be iid random variables with common density f and cumulative distribution function F. Assume $\lim_{x \to F^{-1}(1)} (\frac{d}{dx} \frac{(1-F(x))}{f(x)}) = 0$, then $\lim_{n \to \infty} \mathbb{P}(\max_i X_i \leq a_n x + b_n) = G(x)$ where G is the Gumbel cumulative distribution function and series a_n and b_n are given by $b_n = F^{-1}(1 - \frac{1}{n})$ and $a_n = \frac{1}{nf(b_n)}$.*

Proposition 1 gives a comprehensive description of the law of $\max_i X_i$ needed in our analysis. In our case, we want to characterise the behaviour of $\max_i |X_i|$ where X_i are iid Gaussian random variables.
Let Ψ and ψ be the cdf and density of a standard Gaussian variable X. The cdf of $|X|$ is given by $F = 2\Psi - 1$ and its density is given by $f = 2\psi$ on the positive real line. Thus, $\frac{1-F}{f} = \frac{1-\Psi}{\psi}$ and it is sufficient to verify the condition of Proposition 1 for the standard Gaussian distribution. We have $\lim_{x \to F^{-1}(1)} \frac{d}{dx} \frac{1-\Psi(x)}{\psi(x)} = \lim_{x \to F^{-1}(1)} x \frac{(1-\Psi(x))}{\psi(x)} - 1 = x/x - 1 = 0$, where we have used the fact that $x(1 - \Psi(x)) \sim \phi(x)$ in the large x limit. Let us now find the values of a_n and b_n. In the large x limit, we have

$$\begin{aligned} 1 - F(x) &= 2 \int_x^\infty \frac{e^{-\frac{t^2}{2}}}{\sqrt{2\pi}} dt \\ &= \sqrt{\frac{\pi}{2}} e^{-\frac{x^2}{2}} (\frac{1}{x} + \frac{1}{x^3} + o(\frac{1}{x^3})). \end{aligned}$$

Therefore, one has

$$\log(1 - F(x)) \sim -\frac{x^2}{2}.$$

This yields

$$b_n = F^{-1}(1 - \frac{1}{n}) \sim \sqrt{2 \log n}.$$

Using the same asymptotic expansion of $1 - F(x)$, we can obtain a more precise approximation of b_n

$$b_n = \sqrt{2\log n}\Big(1 - \frac{\log(\log n)}{4\log n} + \frac{\frac{1}{2}\log(\frac{\pi}{4})}{2\log n} - \frac{\log(\log n)}{8(\log n)^2} + o(\frac{\log(\log n)}{(\log n)^2})\Big).$$

Now let us find an approximation for a_n. We have

$$\psi(b_n) \sim \frac{\sqrt{2}}{\pi n}\sqrt{\log n}.$$

Therefore, it follows that

$$a_n \sim \frac{\pi}{\sqrt{2\log n}}.$$

We use these results to lower bound the probability $\mathbb{P}(\max_i |W_i^{l_0}| < \max_i |W_i^k|)$. We have

$$\mathbb{P}(\max_i |W_i^{l_0}| \geq \max_i |W_i^k|) = \mathbb{P}(\max_i |X_i| \geq \gamma_k \max_i |Y_i|),$$

where $\gamma_k = \frac{\alpha_{l_0}}{\alpha_k}$ and (X_i) and (Y_i) are standard Gaussian random variables. Note that $\gamma_k > 1$. Let $A_N = \max_i |X_i|$ and $B_N = \max_i |Y_i|$. We have that

$$\begin{aligned}\mathbb{P}(A_N \geq \gamma_k B_N) &= \mathbb{P}(A_N - \mathbb{E}[A_N] \geq \gamma_k(B_N - \mathbb{E}[B_N]) + \gamma_k\mathbb{E}[B_N] - \mathbb{E}[A_N]) \\ &\leq \mathbb{E}\Big[\frac{(A_N - \mathbb{E}[A_N])^2}{(\gamma_k(B_N - \mathbb{E}[B_N]) + \gamma_k\mathbb{E}[B_N] - \mathbb{E}[A_N]))^2}\Big] \underset{N\to\infty}{\sim} \frac{\pi^2}{4(\gamma_k - 1)^2\log(N)^2}.\end{aligned}$$

We conclude that for large N

$$\mathbb{P}(PR_{l_0}) \geq 1 - \frac{L\pi^2}{4(\gamma - 1)^2\log(N)^2} + o(\frac{1}{\log(N)^2}),$$

where $\gamma = \min_{k\neq l_0} \frac{\alpha_{l_0}}{\alpha_k}$. □

G Additional Experiments

Here we present additional experiments with varying Resnet Architectures (Resnet32/50/104), and sparsities (up to 99.9%) with Relu and Tanh activation functions on Cifar10. We see that overall, using our proposed Stable Resnet performs overall better that standard Resnets.

In addition, we also plot the remaining weights for each layer to get a better understanding on the different pruning strategies and well as understand why some of

the Resnets with Tanh activation functions are untrainable. Furthermore, we added additional MNIST experiments with different activation function (ELU, Tanh) and note that our rescaled version allows us to prune significantly more for deeper networks.

Resnet32	Algo	90	98	99.5	99.9
Relu	SBP-SR	**92.56(0.06)**	**88.25(0.35)**	**79.54(1.12)**	**51.56(1.12)**
	SNIP	92.24(0.25)	87.63(0.16)	77.56(0.36)	10(0)
Tanh	SBP-SR	**90.97(0.2)**	**86.62(0.38)**	**75.04(0.49)**	**51.88(0.56)**
	SNIP	90.69(0.28)	85.47(0.18)	10(0)	10(0)

Resnet50	Algo	95	98	99.5	99.9
Relu	SBP-SR	**92.05(0.06)**	**89.57(0.21)**	**82.68(0.52)**	**58.76(1.82)**
	SNIP	91.64(0.14)	89.20(0.54)	80.49(2.41)	19.98(14.12)
Tanh	SBP-SR	**90.43(0.32)**	**88.18(0.10)**	**80.09(0.0.55)**	**58.21(1.61)**
	SNIP	89.55(0.10)	10(0)	10(0)	10(0)

Resnet104	Algo	95	98	99.5	99.9
Relu	SBP-SR	**93.80(0.13)**	**92.08(0.14)**	**87.47(0.23)**	**72.70(0.48)**
	SNIP	93.18(0.14)	91.47(0.16)	33.63(33.27)	10(0)
Tanh	SBP-SR	**90.43(0.32)**	**88.18(0.10)**	**80.09(0.0.55)**	**58.21(1.61)**
	SNIP	89.55(0.10)	10(0)	10(0)	10(0)

Lastly, for completeness, we also added experiments on the Tiny imagenet dataset for ResNet32 and ResNet50. SBP-SR performs better if not similar to the current state-of-the-art algorithm (GraSP [Wang et al., 2020]). Note, that the differences become more apparent, once we have deeper architectures, which coincides with our theory, as we analysed the case where depth goes to infinity.

Resnet32	Algo	0.85	0.90	0.95
Relu	SBP-SR	**57.25(0.09)**	**55.67(0.21)**	50.63(0.21)
	SNIP	56.92(0.33)	54.99(0.37)	49.48(0.48)
	GraSP	**57.25(0.11)**	55.53(0.11)	**51.34(0.29)**

Resnet50	Algo	0.85	0.90	0.95
Relu	SBP-SR	**59.8(0.18)**	**57.74(0.06)**	**53.97(0.27)**
	SNIP	58.91(0.23)	56.15(0.31)	51.19(0.47)
	GraSP	58.46(NA)	57.48(NA)	52.5(NA)

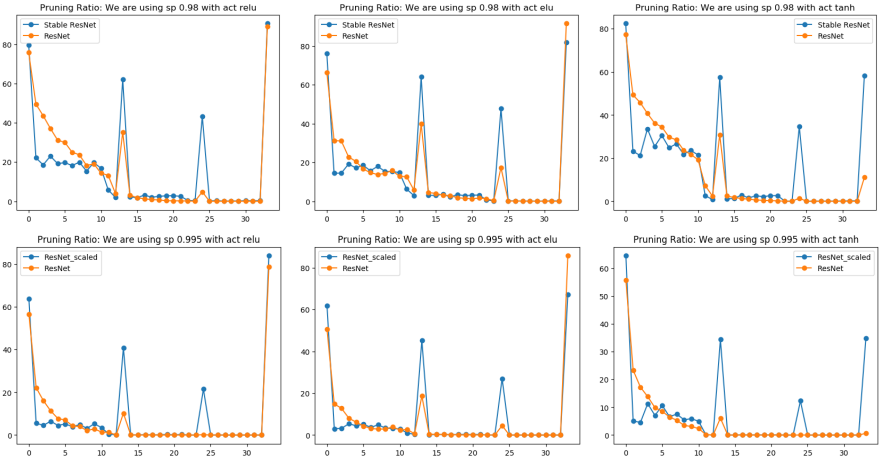
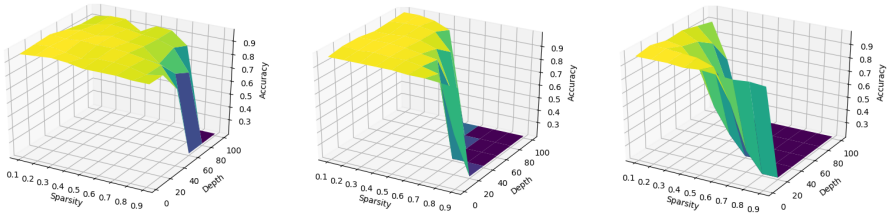
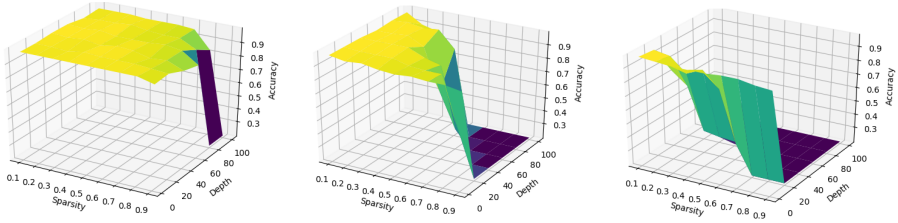


Figure 4: Percentage of pruned weights per layer in a ResNet32 for our scaled ResNet32 and standard Resnet32 with Kaiming initialization



(a) ELU with EOC Init & Rescaling (b) ELU with EOC Init (c) ELU with Ordered phase Init



(d) Tanh with EOC Init & Rescaling (e) Tanh with EOC Init (f) Tanh with Ordered phase Init

Figure 5: Accuracy on MNIST with different initialization schemes including EOC with rescaling, EOC without rescaling, Ordered phase, with varying depth and sparsity. This figure clearly illustrates the benefits of rescaling very deep and sparse FFNN.