On Initialisation of Deep Neural Networks

Alireza Naderi
Mathematical Institute
University of Oxford
naderi@maths.ox.ac.uk

Abstract

Random initialisation of deep neural networks plays an important role in the efficiency of their training. Practical obstacles such as vanishing and exploding gradients and rapid convergence of pairwise input correlations can be avoided by carefully initialising the network on the so-called *Edge of Chaos*. Moreover, the success of sparsely connected neural networks in practice, with <1% of their parameters retained, has given rise to a line of research on pruning the network *at initialisation*, that would be much more cost-effective compared to pruning the trained dense network. We will review the recent theoretical results on the randomly initialised fully-connected networks, as well as pruning-at-initialisation (PaI) literature, and will take a step toward the rigorous analysis of sparse neural networks.

1 Introduction

Although deep neural networks have become the most popular among machine learning models, their training can be quite challenging. Particularly, there are obstacles that lead to the absolute failure of the trained model in achieving an accuracy better than random guessing. Such problems are associated with quantities growing (or shrinking) exponentially with the number of layers, causing numerical instability and inefficient training. These include the problem of exploding and vanishing gradients that has been identified as early as [BFS93] (later discussed in several works, e.g. [Hoc98; PMB13; Han18]), and the problem of rapidly convergent input correlations that has been described and analysed in [BNL04; GB10; Poo+16; Sch+17], among others.

A remedy to such problems can be found in the way the network is initialised. From a dynamical system point of view, with depth representing time, some initial conditions lead to a state of *chaos*, where the network will be oversensitive to the smallest perturbations in the input space, whereas some other initial conditions induce an *ordered* phase, where the network becomes trivial and totally indifferent to the input. Initialisation on the boundary of the two phases, known colloquially as the *edge of chaos*, has been shown to improve the training dynamics, both theoretically and empirically [Poo+16; Sch+17]. Moreover, the instability caused by exploding or vanishing gradients in the backward pass is preventable by maintaining *dynamical isometry* via an appropriate choice of the non-linearity and the initial weight distribution [PSG17; PSG18; MAT22].

A quite different type of modification that can be made at initialisation of the network to enhance the efficiency of its training is *pruning*. It is well-established that extensive pruning of a neural network *after training* does not significantly affect the accuracy (see [Ree93] for an old survey on methods), but also a recent line of research has shown that pruning can even be done right after the weights are randomly initialised and before the training starts. In fact, a random dense neural network is shown to contain very sparse *subnetworks* that can be trained separately and achieve comparable accuracy [FC19]. Of course, such subnetworks

will become more apparent as the training goes on, but if one can identify them before or at the early stages of training, one would save hugely on memory and computation.

We will briefly review the two seemingly disjoint areas of literature regarding initialisation, namely tuning at criticality, and pruning at initialisation. In section 2, we will discuss the problems caused by increasing the depth and the effects of the choice of initialisation parameters on the efficient training of deep neural networks. We will present the criteria introduced in the literature that enables training in large depths. In section 3, we will turn our attention to the sparsely-connected neural networks, focusing on the algorithms proposed in the literature for pruning the dense network either at initialisation or early in the training, as opposed to pruning *after training*. Lastly, in section 4, we will derive criticality conditions for randomly pruned networks in search of a general theoretical analysis of PaI methods.

2 Tuning at criticality

A fully-connected neural network of depth L with the activation function $\sigma : \mathbb{R} \to \mathbb{R}$, maps the input $x_\alpha \in \mathbb{R}^{n_0}$ to the output $z_\alpha^{(L+1)} \in \mathbb{R}^{n_{L+1}}$, through consecutive applications of affine and non-linear transformations, given by

$$z_\alpha^{(\ell+1)} \coloneqq \begin{cases} b^{(\ell+1)} + W^{(\ell+1)}\sigma(z_\alpha^{(\ell)}), & 1 \le \ell \le L \\ b^{(1)} + W^{(1)}x_\alpha, & \ell = 0 \end{cases}. \tag{1}$$

Throughout this article, unless otherwise stated, we will assume that the weights and biases $W^{(\ell+1)} \in \mathbb{R}^{n_{\ell+1}\times n_\ell}$, $b^{(\ell)} \in \mathbb{R}^{n_\ell}$ are randomly initialised as

$$W_{ij}^{(\ell+1)} \sim \mathcal{N}(0, \frac{C_W}{n_\ell}), \quad b_i^{(\ell)} \sim \mathcal{N}(0, C_b) \quad \text{independent}, \tag{2}$$

where C_b and C_W are constants.

The *pre-activations* $z_\alpha^{(\ell+1)}$ converge in distribution to centered Gaussian processes, as the width $n \coloneqq \min\{n_1, \cdots, n_\ell\}$ tends to infinity; a fact known since [Nea96] for single hidden layer networks, and shown for the multi-layer case in [Mat+18; Lee+18]. Thus, in the infinite-width limit, the random map $x_\alpha \mapsto z_{i;\alpha}^{(\ell+1)}$, for each $i \in [n_{\ell+1}]$, is fully described by its covariance function, given by the recursion

$$K^{(\ell+1)}(x_\alpha, x_\beta) = \begin{cases} C_b + C_W\mathbb{E}\big[\sigma(z_{i;\alpha}^{(\ell)})\sigma(z_{i;\beta}^{(\ell)})\big], & 1 \le \ell \le L \\ C_b + \frac{C_W}{n_0}\langle x_\alpha, x_\beta\rangle, & \ell = 0 \end{cases}, \tag{3}$$

where the expectation is taken with respect to the distribution of $z_{i;\cdot}^{(\ell)} \sim \mathsf{GP}(0, K^{(\ell)})$. Therefore, any statistic of interest in an infinitely wide network can be calculated using (3). In particular, the variance of the pre-activations associated with the input x_α can be written as

$$q_\alpha^{(\ell+1)} \coloneqq \mathrm{Var}(z_{i;\alpha}^{(\ell+1)}) = K^{(\ell+1)}(x_\alpha, x_\alpha) = \begin{cases} C_b + C_W\mathbb{E}\big[\sigma\big(\sqrt{q_\alpha^{(\ell)}}Z\big)^2\big], & 1 \le \ell \le L \\ C_b + \frac{C_W}{n_0}\|x_\alpha\|_2^2, & \ell = 0 \end{cases}, \tag{4}$$

where $Z \sim \mathcal{N}(0, 1)$ is a standard normal variable.

We can analyse the layerwise evolution of pre-activations' length by examining the function $V_\sigma(q; C_b, C_W) \coloneqq C_b + C_W\mathbb{E}\big[\sigma\big(\sqrt{q}Z\big)^2\big]$. For monotonic activation functions, the *length map* $V_\sigma : \mathbb{R}_{\ge 0} \to \mathbb{R}_{\ge 0}$ is monotonically increasing and concave [Poo+16]. It is plotted in Figure 1 for tanh activation function where $C_b = 0$. If we choose (σ, C_b, C_W) so that V_σ has a non-zero fixed point, i.e. $\exists q^* > 0$, $V_\sigma(q^*) = q^*$, then there will be a critical radius $r_c^2 \coloneqq \frac{n_0}{C_W}(q^* - C_b)$ such that for inputs x_α of length r_c, the variance of pre-activations does not change with ℓ.

But what happens to the covariance of pre-activations associated with two input signals x_α and x_β, as they propagate through the network? As per (3), the covariance in layer $\ell + 1$ is given by

$$q_{\alpha\beta}^{(\ell+1)} \coloneqq \mathrm{Cov}(z_{i;\alpha}^{(\ell+1)}, z_{i;\beta}^{(\ell+1)}) = C_b + C_W\mathbb{E}\big[\sigma(z_{i;\alpha}^{(\ell)})\sigma(z_{i;\beta}^{(\ell)})\big], \tag{5}$$

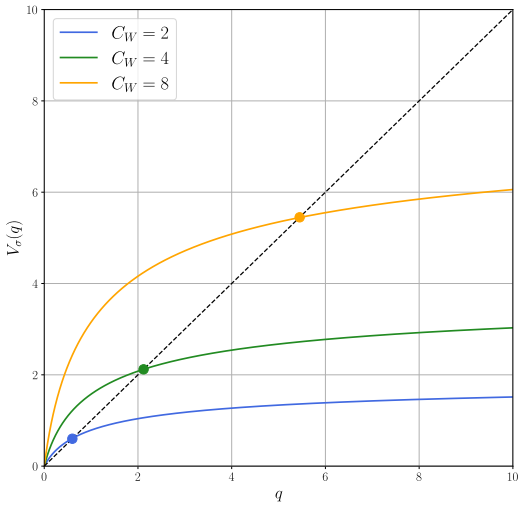

Figure 1: The length map for tanh activation function and no bias has a fixed point $q^* > 0$ when $C_W > 1$.

where $(z_{i;\alpha}^{(\ell)}, z_{i;\beta}^{(\ell)})$ are jointly Gaussian. We can write

$$z_{i;\alpha}^{(\ell)} = \sqrt{q_\alpha^{(\ell)}} Z_1, \quad z_{i;\beta}^{(\ell)} = \sqrt{q_\beta^{(\ell)}} \Big(\rho_{\alpha\beta}^{(\ell)} Z_1 + \sqrt{1 - (\rho_{\alpha\beta}^{(\ell)})^2} Z_2 \Big), \tag{6}$$

where $Z_1, Z_2 \sim \mathcal{N}(0,1)$ are independent, and $\rho_{\alpha\beta}^{(\ell)} \coloneqq q_{\alpha\beta}^{(\ell)} / \sqrt{q_\alpha^{(\ell)} q_\beta^{(\ell)}}$ denotes the correlation at layer ℓ. Assuming the inputs are tuned at critical radius r_c so that $q_\alpha^{(\ell)} = q_\beta^{(\ell)} = q^*$ for all $\ell \geq 0$, we can find a simple *correlation map* $R_\sigma : [-1,1] \to [-1,1]$,

$$R_\sigma(\rho; C_b, C_W) \coloneqq \frac{C_b}{q^*} + \frac{C_W}{q^*} \mathbb{E}\Big[\sigma\big(\sqrt{q^*} Z_1\big) \sigma\big(\sqrt{q^*}(\rho Z_1 + \sqrt{1-\rho^2} Z_2)\big) \Big], \tag{7}$$

so that $\rho_{\alpha\beta}^{(\ell+1)} = R_\sigma(\rho_{\alpha\beta}^{(\ell)})$. Note that $R_\sigma(1) = \frac{C_b + C_W \mathbb{E}[\sigma(\sqrt{q^*} Z)^2]}{q^*} = \frac{V_\sigma(q^*)}{q^*} = 1$, i.e. 1 is a fixed point as expected. Whether it is a stable fixed point, and whether other fixed points exist depend on the choice of (σ, C_b, C_W) [MAT22]. Figure 2 illustrates the correlation map for tanh activation function.

If the choice of (σ, C_b, C_W) dictates that $\rho^* = 1$ is the single stable fixed point, then $\rho_{\alpha\beta}^{(\ell)} \to 1$ with depth, no matter how small the correlation between the two input signals x_α, x_β is. Therefore, in this scenario, all inputs are mapped to a single output, asymptotically with depth. On the other hand, if for a certain choice of (σ, C_b, C_W), there is a single stable fixed point $\rho^* < 1$, then $\rho_{\alpha\beta}^{(\ell)} \to \rho^*$ with depth, for all $x_\alpha \neq x_\beta$, which means two arbitrarily close (but not identical) inputs will be mapped to outputs that are predictably dissimilar. For a given σ, the two regions of the (C_b, C_W) plane that correspond to the two scenarios mentioned above are dubbed *order* and *chaos* regions, respectively. To determine whether $\rho^* = 1$ is an attractive fixed point, [Poo+16; Sch+17] introduce the quantity

$$\chi_1 \coloneqq \left. \frac{\partial \rho_{\alpha\beta}^{(\ell+1)}}{\partial \rho_{\alpha\beta}^{(\ell)}} \right|_{\rho_{\alpha\beta}^{(\ell)} = 1} = R_\sigma'(1) = C_W \mathbb{E}\big[\sigma'(\sqrt{q^*} Z)^2\big]. \tag{8}$$

If the slope χ_1 at $\rho^* = 1$ is less than 1, then the graph of the correlation map lies above the identity, and $\rho^* = 1$ is a stable fixed point, corresponding to the ordered phase. Contrarily, if $\chi_1 > 1$, then this fixed point is unstable and the network is in the chaotic phase, which

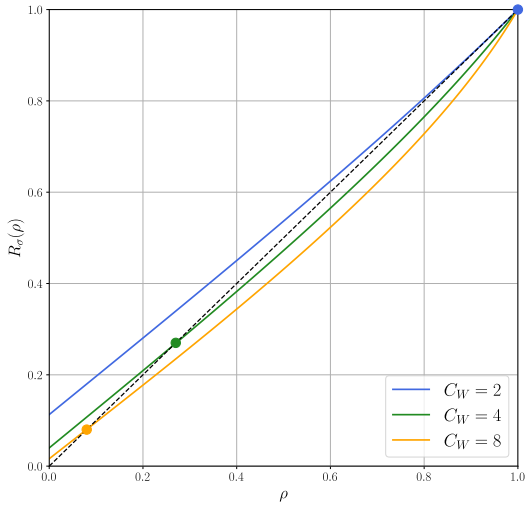

Figure 2: The attractive fixed point of $R_\sigma(\rho)$ for $\sigma(\cdot) = \tanh(\cdot)$ might be 1 (order) or less than 1 (chaos).

means arbitrarily close inputs will separate as they propagate layer by layer. The phase transition boundary $\{(C_b, C_W) \in \mathbb{R}_{\geq 0} \times \mathbb{R}_{\geq 0} \mid \chi_1(C_b, C_W) = 1\}$ is referred to as the *edge of chaos* (EOC), illustrated in Figure 3. As shown in [Sch+17], if the network is not initialised on the EOC, then $\rho^{(\ell)}_{\alpha\beta} \to \rho^*$ exponentially fast with depth, in both ordered ($\rho^* = 1$) and chaotic ($\rho^* \neq 1$) phases. Therefore, to avoid the risk of a network that rapidly correlates or decorrelates the inputs as they propagate through it, one needs to initialise the weights on (or close to) the EOC. Experimental evidence in [Sch+17], also, supports this conclusion, showing that initialisation close to the EOC reduces the training time.

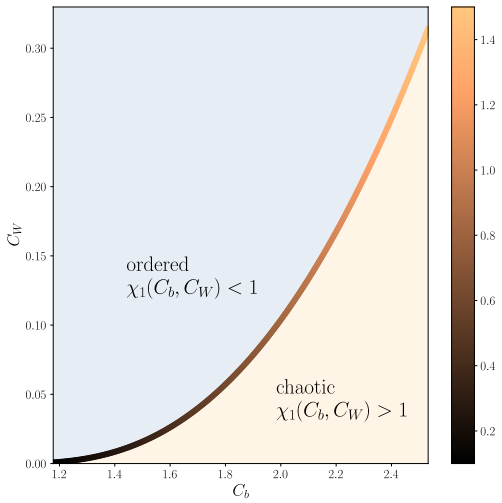

Figure 3: The phase transition diagram with EOC color showing the corresponding q^*.

So far, we have only considered the forward propagation of signals in a deep network. However, the backpropagation of the gradients can be similarly problematic in large depths. With the network defined as in (1), the backpropagation error vector $\delta_\alpha^{(\ell)} \coloneqq \nabla_{z_\alpha^{(\ell)}} \mathcal{C}$ associated with the input x_α is given by the recursion

$$\delta_\alpha^{(\ell)} = \begin{cases} \nabla_{z_\alpha^{(L+1)}} \mathcal{C}, & \ell = L+1 \\ D_\alpha^{(\ell)} (W^{(\ell+1)})^\top \delta_\alpha^{(\ell+1)}, & 1 \leq \ell \leq L \end{cases}, \tag{9}$$

where $\mathcal{C}$ is the cost function and $D_\alpha^{(\ell)} = \mathrm{diag}[\sigma'(z_\alpha^{(\ell)})]$. Unlike the pre-activations $z_\alpha^{(\ell)}$ in the forward pass, the error vectors $\delta_\alpha^{(\ell)}$ do not necessarily follow a normal distribution, even if the network's width tends to infinity. However, a mean-field approximation can still be very useful in the analysis of the backward propagation of gradients. As worked out in [Sch+17], a recurrence formula for the variance $\tilde{q}_\alpha^{(\ell)} \coloneqq \mathrm{Var}(\delta_{i;\alpha}^{(\ell)})$ is given by

$$\tilde{q}_\alpha^{(\ell)} = \frac{n_{\ell+1}}{n_\ell} \tilde{q}_\alpha^{(\ell+1)} \chi_1. \tag{10}$$

Although (10) shows that initialisation on the EOC, where $\chi_1 = 1$, stabilises the variance of the local gradients as they backpropagate through the network, it does not guarantee a well-conditioned backpropagation, as the distribution of $\delta_{i;\alpha}^{(\ell)}$ is not fully described by its second-order moment. To ensure the gradients would not vanish or explode in the backward pass, a stronger condition known as *dynamical isometry* has been introduced in [SMG13]. It requires the overall input-output Jacobian of the network $J \coloneqq \frac{\partial z^{(L+1)}}{\partial x}$ to be near isometry. In our notation, it is straightforward to see

$$J = W^{(L+1)} D^{(L)} W^{(L)} \cdots D^{(1)} W^{(1)} = W^{(L+1)} \prod_{\ell=1}^{L} D^{(\ell)} W^{(\ell)}. \tag{11}$$

If the Jacobian operator is well-conditioned, then the backpropagation operator mapping the output errors to weight matrices at a given layer is also well-conditioned for all layers [PSG17; PSG18]. To that end, the authors of [PSG17; PSG18] consider the entire spectrum of J as a random matrix and require its singular values to concentrate around 1.

Now, for simplicity, suppose that the width is constant throughout the network ($n_\ell = n$ for all $\ell \in [L]$) and there is no affine transformation in the last layer ($W^{(L+1)} = I$, $b^{(L+1)} = 0$). In this case, [PSG17; PSG18] use tools from free probability to derive expressions for the first and second moments of the spectrum of $JJ^\top$, i.e.

$$m_1 = (C_W \mu_1)^L, \tag{12}$$

$$m_2 = (C_W \mu_1)^{2L} L \Big(\frac{\mu_2}{\mu_1^2} + \frac{1}{L} - 1 - s_1 \Big), \tag{13}$$

where μ_k and s_k are the k^{th} moments of the spectra of D^2 and $WW^\top$, respectively. The term $C_W \mu_1 = C_W \mathbb{E}[\sigma'(\sqrt{q^*} Z)^2]$ is exactly χ_1 (see Eq. (8)). Therefore, initialisation on the EOC only guarantees that the Jacobian and the backpropagation local gradients are well-conditioned *on average*, i.e. $m_1 = 1$. Even when $\chi_1 = 1$, the variance of singular value distribution

$$m_2 - m_1^2 = L \Big(\frac{\mu_2}{\mu_1^2} - 1 - s_1 \Big) \tag{14}$$

grows linearly in the depth, unless the activation function and the distribution of weights are carefully chosen so that $\mu_2 / \mu_1^2 \approx 1 + s_1$ [MAT22].

3 Pruning at initialisation

It has long been known that we can discard the majority of the weights of an overparameterised model after training, without a significant loss of test accuracy [LDS89; HS92]. Although pruning the trained neural network reduces the storage cost and the computational cost at

the time of inference [Li+17], it still requires the expensive training of the full model (with millions of parameters) as the starting point. Recent research, however, has shown that for a given randomly initialised fully-connected neural network, there exist sparsely-connected subnetworks (often known as *winning lottery tickets*), that when trained in isolation, match the test accuracy of their respective dense network [FC19]. We will get back to the *lottery ticket hypothesis* in setion 4.

Efficient identification of the winning tickets at initialisation has been the subject of extensive empirical study [LAT19; WZG20; Tan+20; Fra+21]. The deep learning models are so overparameterised that even random pruning does not harm the prediction accuracy up to a certain sparsity level. Therefore, any reasonable pruning-at-initialisation method must significantly outperform the random benchmark. We will take a closer look at several competing methods, namely magnitude pruning, SNIP [LAT19], GraSP [WZG20], and SynFlow [Tan+20].

Suppose a network with vectorised weights $\theta^{(\ell)} \in \mathbb{R}^{p_\ell}$ at each layer $\ell \in [L]$, and a sparsity level $s \in (0,1)$ are given. A pruning method generates binary masks $m^{(\ell)} \in \{0,1\}^{p_\ell}$, $\ell \in [L]$, so that $s = \sum_\ell m^{(\ell)} / \sum_\ell p_\ell$. The pruned network, then, will have weights $m^{(\ell)} \odot \theta^{(\ell)}$ at layer ℓ, where $\odot$ denotes element-wise multiplication. The algorithm first calculates a *salience score* vector $z^{(\ell)} = (z(\theta_1^{(\ell)}), \cdots, z(\theta_{p_\ell}^{(\ell)}))$, and then removes the weights with the smallest scores. This is either done in *one-shot* by setting $m^{(\ell)} = \mathbb{1}(z^{(\ell)} > t)$, where the threshold t is so that the desired sparsity level is met at once; or *iteratively* by repeatedly scoring and pruning the remaining weights to get from sparsity level 0 to s in N steps. In the following section, we will give a brief description of the most remarkable (early or at-initialisation) pruning methods proposed in the literature.

3.1 Methods

Random. There are two slightly different methods called "random pruning" and used as the naive baseline in the literature. One that assigns a uniformly random score $z(\theta_j) \sim \mathsf{Uni}(0,1)$ to each weight and removes those scored under a threshold according to the sparsity level s [Fra+21], and one that draws the entries of the mask $m^{(\ell)}$ independently from a Bernoulli distribution $\mathsf{Bern}(p)$ [Bla+20]. More on the latter to follow in section 4.

Magnitude. In this method, the scores are simply the magnitudes of the weights, i.e. $z(\theta_j) = |\theta_j|$. Those with the lowest scores are then removed at once so that the network attains the sparsity level s. It is a common method for pruning *after training* [Han+15], where the rationale is better justified.

SNIP [LAT19]**.** This method computes the gradient of the loss function using a mini-batch of the training data, assigns the score $z(\theta_j) = |\frac{\partial \mathcal{C}}{\partial \theta_j} \cdot \theta_j|$ to each weight, and discards weights with the lowest scores in single-shot. By doing so, it seeks to preserve the weights that their *connectivity* has the largest effect on the loss. The connectivity parameter c_j for each weight θ_j is allowed to take values in $[0,1]$, rather than $\{0,1\}$. Thus, we can take the derivative of loss with respect to c_j by

$$\left.\frac{\partial \mathcal{C}}{\partial c_j}\right|_{c_j=1} = \lim_{\delta \to 0} \frac{\mathcal{C}(\mathbf{1} \odot \Theta; \mathcal{D}) - \mathcal{C}((\mathbf{1} - \delta e_j) \odot \Theta; \mathcal{D})}{\delta} = \theta_j \frac{\partial \mathcal{C}}{\partial \theta_j}. \tag{15}$$

In other words, SNIP defines the sensitivity of a connection by the change in loss when the weight is modified by a multiplicative factor $1 - \delta$. Several iterative variants of this method are proposed recently, including SNIP-it [VSF20] and FORCE [Jor+21].

GraSP [WZG20]**.** This method computes the Hessian $H = (\frac{\partial^2 \mathcal{C}}{\partial \theta_i \partial \theta_j})_{i,j}$ and the gradient $g = (\frac{\partial \mathcal{C}}{\partial \theta_j})_j$ of the loss function using a mini-batch of the training data, assigns the score $z(\theta_j) = -\theta_j (Hg)_j$, and removes the weights with the lowest scores. The underlying motivation is to prune those weights that their removal will "increase the gradient flow". Since a larger gradient norm roughly results in a larger reduction in the loss at a descent step, GraSP seeks to preserve (or even increase) $\gamma = \|g\|^2$. Note that $\nabla \gamma = 2Hg$, hence the choice of salience score. A variant is proposed in [Fra+21] that substitutes the GraSP score by its magnitude.

SynFlow [Tan+20]**.** This is an iterative method, motivated by avoiding *layer-collapse*, that is when all the weights of a certain layer get pruned and the network becomes disconnected. It is also data-agnostic, unlike the other state-of-the-art methods. SynFlow first replaces all the weights θ_j with their magnitudes $|\theta_j|$, then makes a forward pass of an input of all ones through the network, computes the sum of logits R, and assigns the score $z(\theta_j) = |\frac{\partial R}{\partial \theta_j} \cdot \theta_j|$ to each weight. At each iteration $n \in [N]$, it assigns new scores and prunes the network from sparsity level $s^{\frac{n-1}{N}}$ to the sparsity level $s^{\frac{n}{N}}$.

4 Sparse networks: a deeper look

The *lottery ticket hypothesis*, as first stated in [FC19], predicts the existence of small subnetworks in a randomly initialised neural network that, when trained separately, attain comparable performance to that of the original network. This intriguing phenomenon, backed by plenty of empirical evidence, has motivated two distinct lines of research. On one hand, several algorithms have been proposed for the identification of such winning tickets as early and efficiently as possible (see section 3). On the other hand, the hypothesis itself has been examined in several follow-up papers.

Zhou et al. [Zho+19] show through experiments that there are winning tickets that already perform much better than chance with no training on the data. They propose an algorithm to uncover a subnetwork within the original random network that achieves surprisingly high accuracy (e.g. 86% on MNIST and 41% on CIFAR-10). This is taken further by Ramanujan et al. [Ram+20] by suggesting the "strong lottery ticket hypothesis", which states that "if a neural network with random weights is sufficiently overparameterised, it will contain a subnetwork that performs as well as a trained neural network with the same number of parameters." This conjecture was proven in [Mal+20]. To wit, they prove that the performance of any trained neural network with width n and depth L can be reached by a subnetwork of a randomly-initialised network of width $O(\text{poly}(n, L))$ and depth $2L$, with high probability. The proof was later progressively improved in [OHR20; Pen+20; Bur22], reducing the required width and depth to $O(n \log(nL))$ and $L + 1$, respectively.

Although the existence of lottery tickets is proven in a strong sense, the efficacy of the pruning-at-initialisation methods is not mathematically explained yet. In practice, the score-based pruning followed by retraining is far more popular than pruning a much larger random neural network in search of a strong winning ticket. The following section explores in more detail the most basic PaI method, i.e. random pruning, and establishes the criteria for a well-conditioned initialisation.

4.1 Random pruning

Let us consider the random pruning using a Bernoulli mask. Let the network be as in (1), with the weights at initialisation given by

$$W_{ij}^{(\ell)} = BZ, \tag{16}$$

where $B \sim \mathsf{Bern}(p)$ and $Z \sim \mathcal{N}(0, C_W/n_{\ell-1})$ are drawn independently; and the biases $b_i^{(\ell)} \sim \mathcal{N}(0, C_b)$ as usual. We will first show that under suitable assumptions, the mean-field formalism still describes the pruned network in the infinite width regime.

For a finite network, one can observe that the distribution of the pre-activation at layer $\ell + 1$, i.e. $z_{i;\alpha}^{(\ell+1)}$, conditioned on the last layer, will not be Gaussian anymore. In fact, its conditional density is a mixture of centered Gaussians with the components each having a different variance. More precisely,

$$f_{z_{i;\alpha}^{(\ell+1)}|\mathcal{F}^{(\ell)}}(x) = \sum_{S \subseteq [n_\ell]} p^{|S|}(1-p)^{n_\ell - |S|} \varphi\Big(x;\, 0, C_b + \frac{C_w}{n_\ell} \sum_{j \in S} \sigma(z_{j;\alpha}^{(\ell)})^2\Big), \tag{17}$$

where $\varphi(\cdot;\mu,\sigma^2)$ is the density of a $\mathcal{N}(\mu,\sigma^2)$ distribution. We can explicitly write the conditional moment-generating function as

$$M_{z_{i;\alpha}^{(\ell+1)}}(t)=\exp\Big(\frac{C_b}{2}t^2\Big)\prod_{j=1}^{n_\ell}\Big[1+p\Big(\exp\Big(\frac{C_w\sigma(z_{j;\alpha}^{(\ell)})^2}{2n_\ell}t^2\Big)-1\Big)\Big]. \tag{18}$$

Considering the cumulant generating function $K(t)\coloneqq\log M_{z_{i;\alpha}^{(\ell+1)}}(t)$, we can derive the following formulae for the leading non-zero cumulants at layers $\ell\geq 1$:

$$\kappa_2^{(\ell+1)}=C_b+p\frac{C_w}{n_\ell}\sum_{j=1}^{n_\ell}\sigma(z_{j;\alpha}^{(\ell)})^2, \tag{19}$$

$$\kappa_4^{(\ell+1)}=3p(1-p)\big(\frac{C_w}{n_\ell}\big)^2\sum_{j=1}^{n_\ell}\sigma(z_{j;\alpha}^{(\ell)})^4, \tag{20}$$

$$\kappa_6^{(\ell+1)}=15p(1-p)(1-2p)\big(\frac{C_w}{n_\ell}\big)^3\sum_{j=1}^{n_\ell}\sigma(z_{j;\alpha}^{(\ell)})^6. \tag{21}$$

In general, the odd-ordered cumulants are 0 and the even-ordered ones are of the form

$$\kappa_{2r}^{(\ell+1)}=\begin{cases}C_b+p\frac{C_w}{n_\ell}\sum_{j=1}^{n_\ell}\sigma(z_{j;\alpha}^{(\ell)})^2, & r=1\\ \phi_r(p)\big(\frac{C_w}{n_\ell}\big)^r\sum_{j=1}^{n_\ell}\sigma(z_{j;\alpha}^{(\ell)})^{2r}, & r\geq 2\end{cases}, \tag{22}$$

where ϕ_r is a polynomial of order r, such that $\phi_r(0)=\phi_r(1)=0$. Similarly, for the input layer we have

$$\kappa_{2r}^{(1)}=\begin{cases}C_b+p\frac{C_w}{n_0}\|x_\alpha\|_2^2, & r=1\\ \phi_r(p)\big(\frac{C_w}{n_0}\big)^r\|x_\alpha\|_{2r}^{2r}, & r\geq 2\end{cases}. \tag{23}$$

Equation (22) suggests that the cumulants of order higher than 2 converge to zero in the hidden layers, as the width of the network $n=\min\{n_1,n_2,\cdots,n_L\}$ goes to infinity. Because $\sigma(z_{j;\alpha}^{(\ell)})^{2r}$ does not scale with n_ℓ, then $\kappa_{2r}^{(\ell+1)}$ must decay like $O(n^{1-r})$, thus in the infinite-width limit, the distribution in the hidden layers converges to the Gaussian law.

On the contrary, since the input dimension is fixed, the higher-order cumulants in (23) will not vanish unless $p=1$, i.e. we need to leave the input layer unpruned in order to maintain Gaussianity. We will make this assumption hereafter, that is the input layer remains unpruned and the weights in the subsequent layers get pruned with probability $1-p$. Also, we emphasize that we do not rescale the pre-activations based on the pruning rate (as it is usually done when performing dropout). Under such assumptions, the pre-activations at all layers still converge in distribution to a Gaussian process in the infinite-width limit. Similar to (3), we can write the covariance function of this process at layer ℓ as

$$\overline{K}^{(\ell+1)}(x_\alpha,x_\beta)=\begin{cases}C_b+pC_W\mathbb{E}\big[\sigma(z_{i;\alpha}^{(\ell)})\sigma(z_{i;\beta}^{(\ell)})\big], & 1\leq\ell\leq L\\ C_b+\frac{C_W}{n_0}\langle x_\alpha,x_\beta\rangle, & \ell=0\end{cases}, \tag{24}$$

where the expectation is taken with respect to the distribution of $z_{i;\cdot}^{(\ell)}\sim\mathsf{GP}(0,\overline{K}^{(\ell)})$. Therefore, considering the layerwise recursion for the variance of pre-activations in the pruned network, the length map is given by

$$\overline{V}_\sigma(q;p,C_b,C_W)=C_b+pC_W\mathbb{E}\big[\sigma(\sqrt{q}Z)^2\big], \tag{25}$$

and its fixed point $\overline{q}^*$ is in general different from that of the unpruned network. Similarly, given this new $\overline{q}^*$, the critical radius $\overline{r}_c=\frac{n_0}{C_W}(\overline{q}^*-C_b)$, and the correlation map can be written as

$$\overline{R}_\sigma(\rho;p,C_b,C_W)=\frac{C_b}{\overline{q}^*}+p\frac{C_W}{\overline{q}^*}\mathbb{E}\Big[\sigma\big(\sqrt{\overline{q}^*}Z_1\big)\sigma\big(\sqrt{\overline{q}^*}(\rho Z_1+\sqrt{1-\rho^2}Z_2)\big)\Big], \tag{26}$$

which defines a new EOC where $\overline{\chi}_1=\overline{R}_\sigma'(1)=1$. Here, it is very important to note a subtle difference between our analysis and the analysis of dropout in [Sch+17]. Unlike the dropout

mask considered in [Sch+17], the Bernoulli mask used for random pruning, once drawn, will be fixed for all inputs. Therefore, we do not share their conclusion that dropout will "destroy" the phase transition phenomenon.

Under the assumptions we made, random pruning does not essentially affect the forward criticality conditions. In fact, an infinitely wide randomly pruned network at initialisation, parametrised by (σ, p, C_b, C_W), is equivalent to a fully-connected network with (σ, C_b, pC_W), regarded as a dynamical system in the forward pass. We will leave the study of backpropagation dynamics under random pruning, as well as analysis of other PaI methods to future works.

References

[BFS93] Y. Bengio, P. Frasconi, and P. Simard. "The problem of learning long-term dependencies in recurrent networks". In: *IEEE International Conference on Neural Networks*. 1993, 1183–1188 vol.3. DOI: 10.1109/ICNN.1993.298725.

[BNL04] Nils Bertschinger, Thomas Natschläger, and Robert Legenstein. "At the edge of chaos: Real-time computations and self-organized criticality in recurrent neural networks". In: *Advances in neural information processing systems* 17 (2004).

[Bla+20] Davis Blalock et al. "What is the state of neural network pruning?" In: *Proceedings of machine learning and systems* 2 (2020), pp. 129–146.

[Bur22] Rebekka Burkholz. "Most Activation Functions Can Win the Lottery Without Excessive Depth". In: *Advances in Neural Information Processing Systems*. Ed. by Alice H. Oh et al. 2022. URL: https://openreview.net/forum?id=NySDKS9SxN.

[FC19] Jonathan Frankle and Michael Carbin. "The Lottery Ticket Hypothesis: Finding Sparse, Trainable Neural Networks". In: *International Conference on Learning Representations*. 2019. URL: https://openreview.net/forum?id=rJl-b3RcF7.

[Fra+21] Jonathan Frankle et al. "Pruning Neural Networks at Initialization: Why Are We Missing the Mark?" In: *International Conference on Learning Representations*. 2021. URL: https://openreview.net/forum?id=Ig-VyQc-MLK.

[GB10] Xavier Glorot and Yoshua Bengio. "Understanding the difficulty of training deep feedforward neural networks". In: *Proceedings of the thirteenth international conference on artificial intelligence and statistics*. JMLR Workshop and Conference Proceedings. 2010, pp. 249–256.

[Han+15] Song Han et al. "Learning both weights and connections for efficient neural network". In: *Advances in neural information processing systems* 28 (2015).

[Han18] Boris Hanin. "Which neural net architectures give rise to exploding and vanishing gradients?" In: *Advances in neural information processing systems* 31 (2018).

[HS92] Babak Hassibi and David Stork. "Second order derivatives for network pruning: Optimal brain surgeon". In: *Advances in neural information processing systems* 5 (1992).

[Hoc98] Sepp Hochreiter. "The vanishing gradient problem during learning recurrent neural nets and problem solutions". In: *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems* 6.02 (1998), pp. 107–116.

[Jor+21] Pau de Jorge et al. "Progressive Skeletonization: Trimming more fat from a network at initialization". In: *International Conference on Learning Representations*. 2021. URL: https://openreview.net/forum?id=9GsFOUyUPi.

[LDS89] Yann LeCun, John Denker, and Sara Solla. "Optimal brain damage". In: *Advances in neural information processing systems* 2 (1989).

[Lee+18] Jaehoon Lee et al. "Deep Neural Networks as Gaussian Processes". In: *International Conference on Learning Representations*. 2018.

[LAT19] Namhoon Lee, Thalaiyasingam Ajanthan, and Philip Torr. "SNIP: Single-shot network pruning based on connection sensitivity". In: *International Conference on Learning Representations*. 2019. URL: https://openreview.net/forum?id=B1VZqjAcYX.

[Li+17] Hao Li et al. “Pruning Filters for Efficient ConvNets”. In: *International Conference on Learning Representations*. 2017. URL: https://openreview.net/forum?id=rJqFGTslg.

[Mal+20] Eran Malach et al. “Proving the lottery ticket hypothesis: Pruning is all you need”. In: *International Conference on Machine Learning*. PMLR. 2020, pp. 6682–6691.

[Mat+18] Alexander G de G Matthews et al. “Gaussian Process Behaviour in Wide Deep Neural Networks”. In: *International Conference on Learning Representations*. 2018.

[MAT22] M Murray, V Abrol, and J Tanner. “Activation function design for deep networks: linearity and effective initialisation”. In: *Applied and Computational Harmonic Analysis* 59 (2022), pp. 117–154.

[Nea96] Radford M Neal. *Bayesian Learning for Neural Networks*. Springer, 1996.

[OHR20] Laurent Orseau, Marcus Hutter, and Omar Rivasplata. “Logarithmic pruning is all you need”. In: *Advances in Neural Information Processing Systems* 33 (2020), pp. 2925–2934.

[PMB13] Razvan Pascanu, Tomas Mikolov, and Yoshua Bengio. “On the difficulty of training recurrent neural networks”. In: *International conference on machine learning*. Pmlr. 2013, pp. 1310–1318.

[PSG17] Jeffrey Pennington, Samuel Schoenholz, and Surya Ganguli. “Resurrecting the sigmoid in deep learning through dynamical isometry: theory and practice”. In: *Advances in neural information processing systems* 30 (2017).

[PSG18] Jeffrey Pennington, Samuel Schoenholz, and Surya Ganguli. “The emergence of spectral universality in deep networks”. In: *International Conference on Artificial Intelligence and Statistics*. PMLR. 2018, pp. 1924–1932.

[Pen+20] Ankit Pensia et al. “Optimal lottery tickets via subset sum: Logarithmic over-parameterization is sufficient”. In: *Advances in neural information processing systems* 33 (2020), pp. 2599–2610.

[Poo+16] Ben Poole et al. “Exponential expressivity in deep neural networks through transient chaos”. In: *Advances in neural information processing systems* 29 (2016).

[Ram+20] Vivek Ramanujan et al. “What’s hidden in a randomly weighted neural network?” In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2020, pp. 11893–11902.

[Ree93] Russell Reed. “Pruning algorithms-a survey”. In: *IEEE transactions on Neural Networks* 4.5 (1993), pp. 740–747.

[SMG13] Andrew M Saxe, James L McClelland, and Surya Ganguli. “Exact solutions to the nonlinear dynamics of learning in deep linear neural networks”. In: *arXiv preprint arXiv:1312.6120* (2013).

[Sch+17] Samuel S. Schoenholz et al. “Deep Information Propagation”. In: *International Conference on Learning Representations*. 2017. URL: https://openreview.net/forum?id=H1W1UN9gg.

[Tan+20] Hidenori Tanaka et al. “Pruning neural networks without any data by iteratively conserving synaptic flow”. In: *Advances in neural information processing systems* 33 (2020), pp. 6377–6389.

[VSF20] Stijn Verdenius, Maarten Stol, and Patrick Forré. “Pruning via iterative ranking of sensitivity statistics”. In: *arXiv preprint arXiv:2006.00896* (2020).

[WZG20] Chaoqi Wang, Guodong Zhang, and Roger Grosse. “Picking Winning Tickets Before Training by Preserving Gradient Flow”. In: *International Conference on Learning Representations*. 2020. URL: https://openreview.net/forum?id=SkgsACVKPH.

[Zho+19] Hattie Zhou et al. “Deconstructing lottery tickets: Zeros, signs, and the supermask”. In: *Advances in neural information processing systems* 32 (2019).