A Signal Propagation Perspective for Pruning Neural Networks at Initialization

Namhoon Lee[1], **Thalaiyasingam Ajanthan**[2], **Stephen Gould**[2], **Philip H. S. Torr**[1]

[1]University of Oxford
[2]Australian National University
[1]{namhoon,phst}@robots.ox.ac.uk
[2]{thalaiyasingam.ajanthan, stephen.gould}@anu.edu.au

Abstract

Network pruning is a promising avenue for compressing deep neural networks. A typical approach to pruning starts by training a model and removing unnecessary parameters while minimizing the impact on what is learned. Alternatively, a recent approach shows that pruning can be done at initialization prior to training. However, it remains unclear exactly why pruning an untrained, randomly initialized neural network is effective. In this work, we consider the pruning problem from a signal propagation perspective, formally characterizing initialization conditions that ensure faithful signal propagation throughout a network. Based on singular values of a network's input-output Jacobian, we find that orthogonal initialization enables more faithful signal propagation compared to other initialization schemes, thereby enhancing pruning results on a range of modern architectures and datasets. Also, we empirically study the effect of supervision for pruning at initialization, and show that often unsupervised pruning can be as effective as the supervised pruning. Furthermore, we demonstrate that our signal propagation perspective, combined with unsupervised pruning, can indeed be useful in various scenarios where pruning is applied to non-standard arbitrarily-designed architectures.

1 Introduction

Deep learning has made great strides in machine learning and been applied to various fields from computer vision and natural language processing, to health care and board games [6]. Despite the immense success, however, it remains challenging to deal with the excessive computational and memory requirements of large neural network models. To this end, lightweight models are often preferred, and *network pruning*, a technique to reduce the number of parameters in a network, has been widely employed to compress deep neural networks [2]. Nonetheless, designing pruning algorithms has been lacking rigorous underpinning, because pruning was typically carried out on a pretrained model as a post-processing step, or it has been incorporated within the training procedure.

Recently, Lee et al. [7] have shown that pruning can be done on randomly initialized neural networks, without a laborious pretraining step, at single-shot prior to training (*i.e.*, *pruning at initialization*). They empirically showed that as long as the initial random weights are drawn from a scaled Gaussian (*e.g.*, [1]), their pruning criterion called Connection Sensitivity (CS) can be used to prune deep neural networks, often to an extreme level of sparsity while maintaining good accuracy once trained. However, it remains unclear as to how pruning at initialization is feasible, how it should be understood and whether it can be extended further.

In this work, we first empirically analyze the effect of initialization on their pruning criterion and the result of pruning for fully-connected networks with varying depths and nonlinearities. Precisely,

Preprint. Under review.

even though SNIP [7] is robust to various initialization schemes, it fails catastrophically in some cases. Deeper investigation shows that the failure cases of SNIP corresponds to the cases where the computed CS is unreliable, in that it does not faithfully measure the sensitivity of each connection. This observation leads to a signal propagation perspective for pruning at initialization, by noting that, to ensure faithful CS, it is important to guarantee that the input and error signals propagates with minimal amplification or attenuation in the forward and backward directions, respectively. Such an interpretation enables us to provide a formal characterization of how one should initialize a network to have faithful CS and in turn effective pruning at initialization.

Our signal propagation perspective is inspired by the recent literature on dynamical isometry and mean-field theory [8, 10, 12, 13] where the objective is to understand and ensure faithful signal propagation while training. In this work, we extend this signal propagation perspective to network pruning at initialization by noting that CS is a form of gradient signal. Then, we show that a sufficient condition to ensure faithful CS is *layerwise dynamical isometry*, which is defined as the singular values of the layerwise Jacobians being close to one. Note that this is a stronger condition than *dynamical isometry* (singular values of the input-output Jacobian being concentrated around one), and yet, the initialization method suggested in existing works, *i.e.*, *orthogonal initialization* [8, 14] in fact satisfies layerwise dynamical isometry and can be employed to obtain faithful CS.

Notice, perfect layerwise dynamical isometry cannot always be ensured in the modern networks that have components such as ReLU nonlinearities [8] and/or batch normalization [15]. However, even in such cases, our experiments on various modern architectures (including convolutional, residual and recurrent networks) indicate that CS computed based on orthogonal initialization is robust and results in effective pruning consistently outperforming pruning based on other initializations. This indicates that the signal propagation perspective is not only important to theoretically understand pruning at initialization, but also it improves the results of pruning for a range of networks of practical interest.

Furthermore, this signal propagation perspective for pruning poses another important question: how informative is the error signal computed on random networks, or can we prune neural networks even without supervision? To understand this, we compute CS with different unsupervised surrogate losses and evaluate the pruning results. Interestingly, our results indicate that we can in fact prune networks in an unsupervised manner to extreme sparsity levels without compromising accuracy, and it often compares competitively to pruning with supervision. Moreover, we test if pruning at initialization can be extended to obtain architectures that yield better performance than standard pre-designed architectures with the same number of parameters. In fact, this process, which we call *neural architecture sculpting*, compares favorably against hand-designed architectures, taking network pruning one step further towards neural architecture search.

2 Preliminaries

The ultimate goal of network pruning is to find an architecture that balances between the model complexity and generalization [11]. While doing so, the principle behind conventional approaches is to find unnecessary parameters such that by eliminating them it reduces the complexity of a model while minimizing the impact on what is learned. Naturally, a typical pruning algorithm starts after convergence to a minimum or is performed during training.

This pretraining requirement has been left unattended until a recent work [7], where the authors showed that pruning can be performed on untrained networks at initiailzation prior to training. They proposed a method called SNIP which relies on a new saliency criterion (*i.e.*, CS) that is designed to be computed at initialization, to identify redundant parameters. Specifically, CS is defined as:

$$s_j(\mathbf{w};\mathcal{D}) = \frac{|g_j(\mathbf{w};\mathcal{D})|}{\sum_{k=1}^{m}|g_k(\mathbf{w};\mathcal{D})|}, \qquad \text{where} \quad g_j(\mathbf{w};\mathcal{D}) = \frac{\partial L(\mathbf{c}\odot\mathbf{w};\mathcal{D})}{\partial c_j}\bigg|_{\mathbf{c}=\mathbf{1}}. \tag{1}$$

Here, s_j is the saliency of the parameter j, $\mathbf{w}$ is the network parameters, $\mathbf{c}$ is the auxiliary indicator variables representing the connectivity of network parameters, and $\mathcal{D}$ is a given dataset. Also, g_j is the derivative of the loss L with respect to c_j, which turned out to be an infinitesimal approximation of the change in the loss with respect to removing the parameter j. Based on the above sensitivity, pruning is performed by choosing top-κ (where κ denotes a desired sparsity level) salient parameters.

It is found that SNIP tends to be reliable as long as the sensitivity is measured at initial weights drawn from a layerwise scaled Gaussian (*e.g.*, [1]). However, it remains rather unclear as to what it means to measure saliency scores on untrained neural networks. We investigate this throughout the paper.

3 Probing into the effect of initialization on pruning

In this section, we provide an empirical analysis on the effect of initialization for pruning untrained random neural networks. To this end, our aim is to provide grounds for the signal propagation perspective to pruning random networks, which we will formalize in the next section.

3.1 Problem setup

Consider a fully-connected, feed-forward neural network with weight matrices $\mathbf{W}^l \in \mathbb{R}^{N \times N}$, biases $\mathbf{b}^l \in \mathbb{R}^N$, pre-activations $\mathbf{h}^l \in \mathbb{R}^N$, and post-activations $\mathbf{x}^l \in \mathbb{R}^N$, for $l \in \{1 \dots K\}$ up to K layers. Now, the feed-forward dynamics of a network can be written as,

$$\mathbf{x}^l = \phi(\mathbf{h}^l) \,, \qquad \mathbf{h}^l = \mathbf{W}^l \mathbf{x}^{l-1} + \mathbf{b}^l \,, \tag{2}$$

where $\phi : \mathbb{R} \to \mathbb{R}$ is an elementwise nonlinearity, and the input is denoted by $\mathbf{x}^0$. Given the network configuration, the parameters are initialized by sampling from a probability distribution, typically a zero mean Gaussian with a layerwise variance scaling (*e.g.*, [1, 3, 5]) for enhanced trainability.

As noted in Lee et al. [7], a variance scaling initialization scheme tends to improve pruning results. In essence, a variance scaling initialization scheme re-scales the standard deviation of the distribution from $\sigma \to \frac{\alpha}{\psi_l}\sigma$, where α is a global amplifier and ψ_l a layerwise attenuator that depends on a predefined architecture specification (*e.g.*, fan-in). Notice that in case of a network with layers of the same width, the variance can be controlled by a single scalar $\gamma = \frac{\alpha}{\psi}$ as $\psi_l = \psi$ for all layers l.

Since CS is computed at initialization, we wish to understand the role of initial weights for pruning. In particular, we will look into the effect of varying initialization on linear and tanh MLP networks of layers $K \in \{3, 5, 7\}$ and $N = 100$ on MNIST with $\sigma = 1$ as the default, similar to [12].

3.2 Observations

Our interest is to see the effect of different initialization (*i.e.*, sampling distribution controlled by γ) on the pruning results. Precisely, we initialize a network with different γ, compute CS, prune the network, and train [1]. We first present the generalization errors in Table 1. Note, even though the training policy is the same and reliable across all experiments, depending on γ, the results can vary critically (*e.g.*, the failure case of tanh network; $K = 7$, $\gamma = 1.0$). This indicates that a different sampling distribution results in a different subnetwork structure or topology as a result of pruning.

To further investigate, we visualize the resulting pruning patterns $\mathbf{c}$ of each layer as well as the corresponding CS used for pruning in Figure 1. It is seen in the sparsity patterns that for the tanh case, unlike the linear case, more number of parameters tend to be pruned in the later layers than the earlier layers. This becomes critical to learning when a high sparsity level is requested; *e.g.*, for $\bar{\kappa} = 90\%$, only a few parameters in later layers are retained after pruning, severely limiting the learning capability of the subnetwork. This is explained by the CS plot. The sensitivity of parameters in the tanh network tends to decrease towards the later layers, and therefore, by choosing the top-κ parameters globally based on CS scores, it will result in a subnetwork in which retained parameters are distributed highly non-uniformly and sparsely throughout the network.

We posit that this unreliability of CS is due to the poor signal propagation: an initialization that leads the forward signal to explode (by a large γ) will saturate the backward signal, *i.e.*, gradients, and since CS is directly related to the gradients, such initialization will result in unreliable CS, and thereby poor pruning results, even completely disconnecting the inputs and outptus. Note that while it becomes worse for tanh networks as the depth K increases, this is not the case for linear networks because a linear network can be essentially represented as a single layer network regardless of the depth. In the following section, we formalize this signal propagation perspective and recommend an initialization that would ensure CS to be reliable.

[1] In order to focus on the effect of initialization purely on pruning, for this experiment only, we re-initialize pruned networks using [1] before training. This way, potential influence from different initial weights to training and thereby generalization errors are controlled, while the subnetworks are well-initialized for stable training.

Table 1: Effect of initialization on generalization errors of pruned networks. Linear and tanh MLP networks of different depths K are initialized with varying $\gamma \in \{10^{-4}, .., 10^{1}\}$, pruned for the sparsity level $\bar{\kappa} = 90\%$, and trained the same standard way. While tanh networks achieve lower generalization errors than linear ones in general (due to their ability to learn a non-linear function, unsurprisingly), there are critical <u>failure cases</u> for some tanh cases, indicating that the topology of subnetworks after pruning can vary significantly depending on the initialization before pruning.

	linear						tanh					
K	10^{-4}	10^{-3}	10^{-2}	10^{-1}	10^{0}	10^{1}	10^{-4}	10^{-3}	10^{-2}	10^{-1}	10^{0}	10^{1}
3	7.99	8.00	7.72	7.69	7.69	7.69	2.54	2.50	2.52	2.50	2.57	2.59
5	8.21	8.15	8.06	8.02	8.01	8.01	2.55	2.53	2.53	2.51	2.45	<u>90.2</u>
7	8.30	8.27	8.37	8.44	8.43	8.44	2.66	2.70	2.67	2.61	<u>90.2</u>	<u>90.2</u>

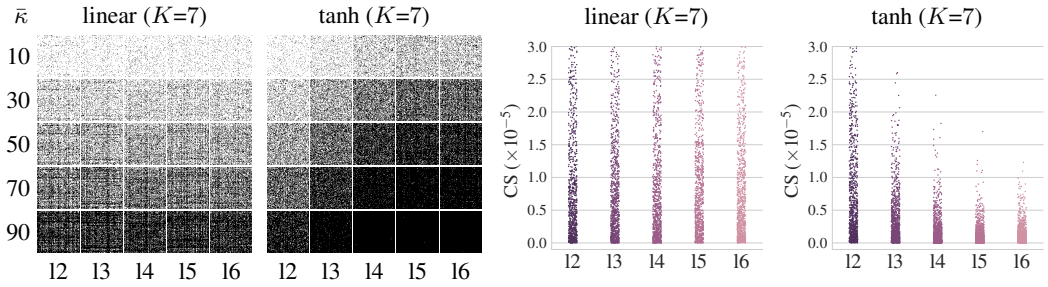

Figure 1: (left) Pruning patterns $c \in \{0, 1\}^{100\times100}$ of each layer: black(0)/white(1) pixels refer to pruned/retained parameters; (right) connection sensitivities measured for the parameters in each layer. Networks are initialized with $\gamma = 1.0$. Unlike the linear case, the sparsity pattern for the tanh network is non-uniform over different layers. This becomes critical when pruning for a high sparsity level; for example, there are only a few parameters retained in later layers for $\bar{\kappa} = 90\%$, which leads to poor learning capability and accounts for critical failure cases observed in Table 1. This is explained by the connection sensitivity plot. For tanh networks, parameters in later layers tend to have lower connection sensitivities compared to parameters in earlier layers.

4 Pruning random networks as deep information propagation

We now provide our signal propagation perspective for network pruning inspired by the recent literature on dynamical isometry and mean-field theory [8, 10, 12, 13]. Especially, we are interested in understanding the forward and backward signal propagation of a neural network at a given initialization and derive properties on the initialization that would ensure that the computed CS is reliably representing the sensitivity of each connection.

4.1 Signal propagation perspective of neural networks

Note that, signal propagation in a network is said to be *faithful* if the input signal is propagated to the output with minimal amplification or attenuation in any of its dimensions. In fact, this faithfulness is directly related to the singular value distribution of the input-output Jacobian of the network. To understand this, we first interpret gradients in terms of layerwise Jacobians and derive conditions on the Jacobians to ensure faithful signal propagation.

4.1.1 Gradients in terms of Jacobians

We consider the problem setup in Section 3.1 where the input-output Jacobian corresponding to a given input $\mathbf{x}^0$ can be written as:

$$\mathbf{J}^{0,K} = \frac{\partial \mathbf{x}^K}{\partial \mathbf{x}^0} = \prod_{l=1}^{K} \mathbf{D}^l \mathbf{W}^l \,, \tag{3}$$

where $\mathbf{D}^l \in \mathbb{R}^{N\times N}$ is a diagonal matrix with entries $\mathbf{D}^l_{ij} = \phi'(h^l_i)\delta_{ij}$, with ϕ' denoting the derivative of nonlinearity ϕ, and $\delta_{ij} = \mathbb{1}[i = j]$ is the Kronecker delta. Here, we use $\mathbf{J}^{k,l}$ to denote the Jacobian from layer k to layer l. Now, we give the relationship between gradients and the Jacobians below.

Proposition 4.1. Let $\epsilon = \partial L/\partial \mathbf{x}^K$ denote the error signal and $\mathbf{x}^0$ denote the input signal. Then,

1. the gradients satisfy:
$$\mathbf{g}_{\mathbf{w}^l}^T = \epsilon\, \mathbf{J}^{l,K} \mathbf{D}^l \otimes \mathbf{x}^{l-1} \ , \tag{4}$$
where $\mathbf{J}^{l,K} = \partial \mathbf{x}^K/\partial \mathbf{x}^l$ is the Jacobian from layer l to the output and $\otimes$ is the Kronecker product.
2. additionally, for linear networks, *i.e.*, when ϕ is the identity:
$$\mathbf{g}_{\mathbf{w}^l}^T = \epsilon\, \mathbf{J}^{l,K} \otimes \left(\mathbf{J}^{0,l-1}\mathbf{x}^0 + \mathbf{a}\right) \ , \tag{5}$$
where $\mathbf{J}^{0,l-1} = \partial \mathbf{x}^{l-1}/\partial \mathbf{x}^0$ is the Jacobian from the input to layer $l-1$ and $\mathbf{a} \in \mathbb{R}^N$ is the constant term that does not depend on $\mathbf{x}^0$.

Proof. This can be proved by an algebraic manipulation of chain rule while using the feed forward dynamics (2). We provide the full derivation in Appendix A. □

Notice that, the gradient at layer l constitutes both the backward propagation of the error signal ϵ up to layer l and the forward propagation of the input signal $\mathbf{x}^0$ up to layer $l-1$. Moreover, especially in the linear case, the signal propagation in both directions is governed by the corresponding Jacobians. We believe, this interpretation of gradients is useful as it sheds light on how signal propagation affects the gradients. To this end, next we analyze the conditions on the Jacobians which would guarantee faithful signal propagation and consequently faithful gradients.

4.1.2 Ensuring faithful gradients

Here, we first consider the layerwise signal propagation which would be useful to derive properties on the initialization to ensure faithful gradients. To this end, let us consider the layerwise Jacobian:

$$\mathbf{J}^{l-1,l} = \frac{\partial \mathbf{x}^l}{\partial \mathbf{x}^{l-1}} = \mathbf{D}^l \mathbf{W}^l \ . \tag{6}$$

Note that, to ensure faithful signal propagation, it is sufficient to ensure *layerwise dynamical isometry* which is defined as the singular values of the layerwise Jacobian $\mathbf{J}^{l-1,l}$ being close to one. This would guarantee that the signal from layer l to $l-1$ (or vice versa) is propagated without amplification or attenuation in any of its dimension. From Proposition 4.1, by induction, it is easy to show that if the layerwise signal propagation is faithful, the error and input signals are faithfully propagated throughout the network, resulting in faithful gradients.

For a linear neural network, $\mathbf{J}^{l-1,l} = W^l$. Therefore one can initialize the weight matrices to be *orthogonal*, meaning for each layer l, initialize the weight matrix such that $(\mathbf{W}^l)^T\mathbf{W}^l = \mathbf{I}$, where $\mathbf{I}$ is the identity matrix of dimension N. In this case, all the singular values of $\mathbf{W}^l$ are exactly one and such an initialization would guarantee faithful gradients.

On the other hand, the case of nonlinear neural networks is more complicated as the diagonal matrix $\mathbf{D}^l$ depends on the pre-activations at layer l. In this case, the intuitive idea is to make sure that the pre-activations $\mathbf{h}^l$ falls into the linear region of the nonlinear function ϕ. Precisely, *mean-field theory* [13] assumes that for large-N limit, the empirical distribution of the pre-activations $\mathbf{h}^l$ converges to a Gaussian with zero mean and variance q^l, where the variance follows a recursion relation. Therefore, to ensure layerwise faithful signal propagation, the idea is to find the fixed point q^* such that $\mathbf{h}^l \sim \mathcal{N}(0, q^*)$ for all $l \in \{1 \dots K\}$. Such a fixed point makes $\mathbf{D}^l = \mathbf{D}$ for all layers, ensuring that the pre-activations are indeed in the linear region of the nonlinearity. In this case, given the nonlinearity, one can find a rescaling such that $(\mathbf{D}\mathbf{W}^l)^T(\mathbf{D}\mathbf{W}^l) = (\mathbf{W}^l)^T\mathbf{W}^l/\sigma_w^2 = \mathbf{I}$. The procedure for finding the rescaling σ_w^2 for various nonlinearties are discussed in [8, 9].

Note that *dynamical isometry* (singular values of the input-output Jacobian being concentrated around one) is regarded as the condition for faithful signal propagation in a network [8, 12]. In fact, this is a weaker condition than our layerwise dynamical isometry. However, in practice, the initialization method suggested in the existing works [8, 14], *i.e.*, *orthogonal initialization for weight matrices in each layer with mean-field theory based rescaling*, satisfy layerwise dynamical isomtery, even though this term was not mentioned.

Table 2: Jacobian singular values and resulting sparse networks for 7-layer tanh MLP network considered in Section 3. Note that the failure cases correspond to unreliable CS resulted from poorly conditioned initial Jacobians. SG, CN, and Sparsity refer to Scaled Gaussian, Condition Number (*i.e.*, $s_{\max}/s_{\min}$, where $s_{\max}$ and $s_{\min}$ are the maximum and minimum Jacobian singular values), and a ratio of pruned prameters to total number of parameters, respectively.

	Jacobian singular values			Sparsity in pruned network	
Initialization	Mean	Std	CN	l1 | l2 | l3 | l4 | l5 | l6 | l7	**Error**
SG (γ=10^{-4})	2.46e−07	9.90e−08	4.66e+00	0.97 | 0.80 | 0.80 | 0.80 | 0.80 | 0.81 | 0.48	2.66
SG (γ=10^{-3})	5.74e−04	2.45e−04	8.54e+00	0.97 | 0.80 | 0.80 | 0.80 | 0.80 | 0.81 | 0.48	2.67
SG (γ=10^{-2})	4.49e−01	2.51e−01	5.14e+01	0.96 | 0.80 | 0.80 | 0.80 | 0.81 | 0.81 | 0.49	2.67
SG (γ=10^{-1})	2.30e+01	2.56e+01	2.92e+04	0.96 | 0.81 | 0.82 | 0.82 | 0.82 | 0.80 | 0.45	2.61
SG (γ=10^{0})	1.03e+03	2.61e+03	3.34e+11	0.85 | 0.88 | 0.99 | 1.00 | 1.00 | 1.00 | 0.91	90.2
SG (γ=10^{1})	3.67e+04	2.64e+05	inf	0.84 | 0.95 | 1.00 | 1.00 | 1.00 | 1.00 | 1.00	90.2

4.2 Ensuring faithful connection sensitivity

So far, we have discussed how one should initialize a network to have faithful gradients based on signal propagation. Now, we give the relationship between CS and the gradients and show that the same conditions are sufficient to guarantee faithful CS. Note that, from Eq. (1), CS is a normalized magnitude of gradients with respect to the connectivity parameters $\mathbf{c}$. Here, we use the vectorized notation where $\mathbf{w}$ denotes all learnable parameters and $\mathbf{c}$ denotes the corresponding connectivity parameters. From chain rule, we can write:

$$\left.\frac{\partial L(\mathbf{c}\odot\mathbf{w};\mathcal{D})}{\partial \mathbf{c}}\right|_{\mathbf{c}=\mathbf{1}} = \left.\frac{\partial L(\mathbf{c}\odot\mathbf{w};\mathcal{D})}{\partial (\mathbf{c}\odot\mathbf{w})}\right|_{\mathbf{c}=\mathbf{1}} \odot \mathbf{w} = \frac{\partial L(\mathbf{w};\mathcal{D})}{\partial \mathbf{w}} \odot \mathbf{w}\,. \tag{7}$$

Therefore, $\partial L/\partial \mathbf{c}$ is simply the gradients $\partial L/\partial \mathbf{w}$ amplified (or attenuated) by the corresponding weights $\mathbf{w}$, *i.e.*, $\partial L/\partial c_j = \partial L/\partial w_j\, w_j$ for all $j \in \{1 \dots m\}$. Considering $\partial L/\partial c_j$ for a given j, since w_j does not depend on any other layers or signal propagation, the only term that depends on signal propagation of the network is the gradient term $\partial L/\partial w_j$. Therefore, a necessary condition to ensure faithful $\partial L/\partial \mathbf{c}$ (and CS) is that the gradients $\partial L/\partial \mathbf{w}$ need to be faithful.

Another important point to consider is that, in SNIP [7], a global threshold is chosen based on CS to obtain the pruned network (refer Section 2), which requires the CS scores to be of the same scale for each layer. To this end, for faithful CS, we require the gradients to be faithful and the weights to be in the same scale for all the layers. Notice, this condition is trivially satisfied when the layerwise dynamical isometry is ensured, as each layer is initialized identically (*i.e.*, orthogonal initialization) and the gradients are guaranteed to be faithful.

Note that, based on this signal propagation perspective for network pruning, we explain the failure cases of pruning considered in Section 3. To this end, we measure the singular value distribution of the input-output Jacobian ($\mathbf{J}^{0,K}$) for the the 7-layer, fully-connected tanh network and the results are reported in Table 2. Note that, while CS based pruning is robust to moderate changes in the Jacobian singular values, it failed catastrophically when the condition number of the Jacobian is very large ($>$ 1e+11). In fact, these failure cases correspond to completely disconnected networks as a consequence of pruning based on unreliable CS resulted from poorly conditioned initial Jacobians. As we will show subsequently, these findings extend to modern architectures, and our recommended orthogonal initialization yields well-conditioned Jacobians and in turn the best pruning results.

5 Validation and extensions

In this section, we evaluate the idea of employing faithful initialization on a wide variety of settings. We further study the role of supervision under the pruning at initialization regime, extending it to unsupervised pruning. Our results show that indeed, pruning can be approached from the signal propagation perspective at varying scale, bringing forth the notion of neural architecture sculpting. All experiments were conducted using TensorFlow, and the experiment settings used to generate the presented results are described in detail in Appendix B.

Table 3: Pruning results for various neural networks on different datasets. All networks are pruned for the sparsity $\bar{\kappa} = 90\%$. We report condition numbers (CN) and generalization errors on MNIST (LeNet, GRU) and CIFAR-10 (VGG16, ResNets). (d) and (s) refer to dense and sparse networks, corresponding to the network before and after pruning, respectively. The lowest (or closest to 1) condition numbers and **best** errors are highlighted in each column.

		LeNet			GRU			VGG16		
Initialization	**Superv.**	CN (d)	CN (s)	Error	CN (d)	CN (s)	Error	CN (d)	CN (s)	Error
VS-L [5]	✓	6.7	8.1	1.69	4.8	4.8	1.25	3.7	5.7	8.16
VS-G [1]	✓	6.7	9.4	1.75	3.9	3.9	1.20	3.7	5.0	8.18
VS-H [3]	✓	6.7	9.6	1.81	5.4	5.7	1.28	3.7	11.4	8.36
Orthogonal	✓	4.5	5.7	**1.61**	3.3	3.4	**1.18**	2.9	4.2	**8.11**
Orthogonal	✗	4.5	7.9	1.83	3.3	3.6	1.35	2.9	9.1	8.25

		ResNet32			ResNet56			ResNet110		
Initialization	**Superv.**	CN (d)	CN (s)	Error	CN (d)	CN (s)	Error	CN (d)	CN (s)	Error
VS-L [5]	✓	10.4	50.9	11.96	35.3	185.3	10.43	66.2	1530.9	9.13
VS-G [1]	✓	11.0	55.0	11.89	36.9	191.1	10.60	74.0	1450.0	9.17
VS-H [3]	✓	14.6	234.9	12.21	78.7	2489.4	10.63	182.7	150757.4	9.08
Orthogonal	✓	8.1	36.3	**11.55**	31.3	170.4	**10.08**	83.1	1176.7	8.88
Orthogonal	✗	8.1	38.4	11.69	31.3	186.1	11.01	83.1	1081.4	**8.82**

5.1 Evaluation on various neural networks and datasets

In this section, we validate our signal propagation perspective for pruning networks at initialization. To this end, we demonstrate that the orthogonal initialization enables faithful signal propagation, yielding enhanced pruning results. Specifically, we provide condition numbers of the input-output Jacobian of the networks (*i.e.*, $s_{\max}/s_{\min}$) for both before and after pruning, and show that they correlate highly with the performance of pruned sparse networks. We evaluate various network models on different image classification datasets. All results are the average of multiple runs (*e.g.*, 10 runs for MNIST and 5 runs for CIFAR-10), and we do not optimize anything specific for a particular case (see Appendix B for experiment settings). The results are presented in Table 3.

First of all, the best pruning results are achieved with the orthogonal initialization. Looking it closely, it is evident that there exists a high correlation between the condition numbers (both before and after pruning) and the performance of pruned networks; *i.e.*, the network initialized or pruned to have the lowest condition number achieves the **best** generalization error (with an exception for ResNet110). Note, all Jacobian singular values being close to 1 (*i.e.*, dynamical isometry), by definition, states how faithful a network will be with regard to letting signals to propagate without amplifying or attenuating. Therefore, the fact that the condition number of the dense network being close to 1 (or relatively closer towards 1) tends to yield good generalization errors, validates that our signal propagation perspective is indeed effective for pruning at initialization.

Furthermore, in Table 3, there is a clear correlation between condition numbers of the Jacobians before and after pruning based on CS, *i.e.*, the lower CN of the dense network leads to lower CN of the sparse network in most cases. This clearly indicates that CS is not drastically destroying the signal propagation properties of the network while pruning, even though this is not explicitly enforced. To validate this, we performed random pruning on LeNet with orthogonal initialization for the same sparsity level. Indeed, even though initial Jacobian is well-conditioned, random pruning completely destroyed the signal propagation yielding a sparse network with an extremely large condition number and the higher generalization error of 2.67%.

We further evaluate for wide residual networks on Tiny-ImageNet, and find that the results are consistent (see Table 4). We note that Tiny-Imagenet is in general harder than the original ImageNet, due to the reduced resolution and the number of examples but still a large number of classes. Moreover, pruning of very deep residual networks (*e.g.*, 110 layers) is rarely tested in the literature.

Table 4: Tiny-ImageNet results.

Initialization	**WRN-16**	**WRN-22**
VS-L [5]	45.08 (0.345)	44.20 (1.006)
VS-G [1]	44.56 (0.339)	43.11 (0.274)
VS-H [3]	46.62 (0.684)	44.77 (0.461)
Orthogonal	**44.20** (0.314)	**42.53** (0.348)

5.2 Pruning without supervision

So far, we have shown that pruning random networks can be approached from a signal propagation perspective by ensuring faithful CS. Notice, another factor that constitutes CS is the loss term. At a glance, it is not obvious how informative the supervised loss measured on a random network will be for CS. In this section, we check the effect of supervision, by simply replacing the loss computed using ground-truth labels with different unsupervised surrogate losses as follows: replacing the target distribution using ground-truth labels with uniform distribution (Unif.), and using the averaged output prediction of the network (Pred.; softmax/raw). The results on MLP networks are in Table 5. Even though unsupervised pruning results are not as good as the supervised case, the results are still interesting, especially for the uniform case, in that there was no supervision given to compute CS. We thus experiment the uniform case on other networks as well (see Supervision=✗ in Table 3). Surprisingly, unsupervised pruning is not significantly worse than the supervised, and often yields competitive results (*e.g.*, ResNet110). Notably, previous pruning algorithms premise the existance of supervision as a priori. Being the first demonstration of unsupervised pruning, along with the signal propagation perspective, one could find it useful to apply this unsupervised pruning strategy to scenarios where it lacks labels or is only provided with weak supervision.

Table 5: Unsupervised pruning results. For all cases orthogonal initialization is used, and pruned for the sparsity of $\bar{\kappa} = 90\%$.

		Depth K		
Loss	**Superv.**	3	5	7
GT	✓	2.46	2.43	2.61
Pred. (raw)	✗	3.31	3.38	3.60
Pred. (softmax)	✗	3.11	3.37	3.56
Unif.	✗	2.77	2.77	2.94

5.3 Neural architecture sculpting

We have shown that the signal propagation perspective enables pruning of networks at initialization, even without supervision. This sparks curiosity of whether pruning needs to be limited to pre-shaped architectures. In other words, what if pruning starts with a bulky network and is treated as *sculpting* an archiecture? To seek for an answer, we conduct the following experiments: we take a popular pre-designed architecture (ResNet20) as a base network, and consider a range of variants that are originally bigger than the base model, but pruned to have the same number of parameters as the base dense network. Specifically, we consider the following *equivalents*: **(1)** the same number of residual blocks, but with larger widths; **(2)** a reduced number of residual blocks with larger widths; **(3)** a larger residual block and the same width. The results are presented in Figure 2. Overall, the sparse equivalents record lower generalization errors than the dense base model. Notice that some models are pruned to extreme sparsity levels (*e.g.*, Equivalent 1 pruned for $\bar{\kappa} = 98.4\%$). This result is well aligned with recent research findings in [4]: large sparse networks outperform the small dense network counterpart, and with a dedicated implementation for sparsity, large sparse models can even enjoy a similar computational efficiency as the small dense model in practice. We further note that unlike previous works, the sparse networks are discovered by sculpting, without pretraining nor supervision.

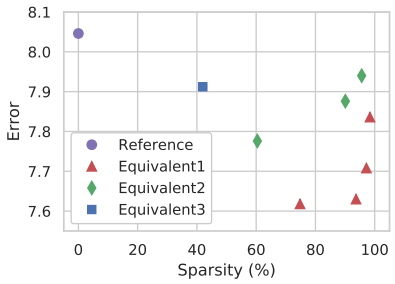

Figure 2: Neural architecture sculpting. While all have the same number of parameters, sparse networks outperform the referece dense network. Errors on CIFAR-10 (average over 5 runs).

6 Discussion

In this work, we have approached network pruning from a signal propagation perspective and formally characterized initialization conditions for a network to be pruned prior to training. Based on this, we found in our experiments that orthogonal initialization produced the best pruning results compared to other initialization schemes on various modern architectures. While pruning on orthogonal initialization empirically produces trainable sparse networks, it would be immensely beneficial if there is a way to guarantee that. We believe, our signal propagation perspective provides a means to formulate this as an optimization problem by maximizing the "trainability" of sparse networks (measured by the Jacobian singluar values) while pruning, and we intend to explore this direction as a future work.

Acknowledgments

This work was supported by the ERC grant ERC-2012-AdG 321162-HELIOS, EPSRC grant Seebibyte EP/M013774/1, EPSRC/MURI grant EP/N019474/1 and the Australian Research Council Centre of Excellence for Robotic Vision (project number CE140100016). We would also like to acknowledge the Royal Academy of Engineering and FiveAI, and thank Richard Hartley and Puneet Dokania for helpful discussions.

References

[1] X. Glorot and Y. Bengio. Understanding the difficulty of training deep feedforward neural networks. *AISTATS*, 2010.

[2] S. Han, H. Mao, and W. J. Dally. Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding. *ICLR*, 2016.

[3] K. He, X. Zhang, S. Ren, and J. Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. *ICCV*, 2015.

[4] N. Kalchbrenner, E. Elsen, K. Simonyan, S. Noury, N. Casagrande, E. Lockhart, F. Stimberg, A. v. d. Oord, S. Dieleman, and K. Kavukcuoglu. Efficient neural audio synthesis. *ICML*, 2018.

[5] Y. LeCun, L. Bottou, G. B. Orr, and K.-R. Müller. Efficient backprop. *Neural Networks: Tricks of the Trade*, 1998.

[6] Y. LeCun, Y. Bengio, and G. Hinton. Deep learning. *Nature*, 2015.

[7] N. Lee, T. Ajanthan, and P. H. Torr. Snip: Single-shot network pruning based on connection sensitivity. *ICLR*, 2019.

[8] J. Pennington, S. Schoenholz, and S. Ganguli. Resurrecting the sigmoid in deep learning through dynamical isometry: theory and practice. *NIPS*, 2017.

[9] J. Pennington, S. S. Schoenholz, and S. Ganguli. The emergence of spectral universality in deep networks. *AISTATS*, 2018.

[10] B. Poole, S. Lahiri, M. Raghu, J. Sohl-Dickstein, and S. Ganguli. Exponential expressivity in deep neural networks through transient chaos. *NIPS*, 2016.

[11] R. Reed. Pruning algorithms-a survey. *Neural Networks*, 1993.

[12] A. M. Saxe, J. L. McClelland, and S. Ganguli. Exact solutions to the nonlinear dynamics of learning in deep linear neural networks. *ICLR*, 2014.

[13] S. S. Schoenholz, J. Gilmer, S. Ganguli, and J. Sohl-Dickstein. Deep information propagation. *ICLR*, 2017.

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

[15] G. Yang, J. Pennington, V. Rao, J. Sohl-Dickstein, and S. S. Schoenholz. A mean field theory of batch normalization. *ICLR*, 2019.

A Gradients in terms of Jacobians

Proposition A.1. Let $\epsilon = \partial L/\partial \mathbf{x}^K$ denote the error signal and $\mathbf{x}^0$ denote the input signal. Then,

1. the gradients satisfy:
$$\mathbf{g}_{\mathbf{w}^l}^T = \epsilon\, \mathbf{J}^{l,K} \mathbf{D}^l \otimes \mathbf{x}^{l-1}\,, \tag{8}$$
where $\mathbf{J}^{l,K} = \partial \mathbf{x}^K/\partial \mathbf{x}^l$ is the Jacobian from layer l to the output and $\otimes$ is the Kronecker product.
2. additionally, for linear networks, *i.e.*, when ϕ is the identity:
$$\mathbf{g}_{\mathbf{w}^l}^T = \epsilon\, \mathbf{J}^{l,K} \otimes \left(\mathbf{J}^{0,l-1}\mathbf{x}^0 + \mathbf{a}\right)\,, \tag{9}$$
where $\mathbf{J}^{0,l-1} = \partial \mathbf{x}^{l-1}/\partial \mathbf{x}^0$ is the Jacobian from the input to layer $l-1$ and $\mathbf{a} \in \mathbb{R}^N$ is the constant term that does not depend on $\mathbf{x}^0$.

Proof. The proof is based on a simple algebraic manipulation of the chain rule. However, we give it here for completeness. From the chain rule, the gradient of the loss with respect to the weight matrix $\mathbf{W}^l$ can be written as:
$$\mathbf{g}_{\mathbf{w}^l} = \frac{\partial L}{\partial \mathbf{W}^l} = \frac{\partial L}{\partial \mathbf{x}^K}\frac{\partial \mathbf{x}^K}{\partial \mathbf{x}^l}\frac{\partial \mathbf{x}^l}{\partial \mathbf{W}^l}\,. \tag{10}$$
Here, the gradient $\partial \mathbf{y}/\partial \mathbf{x}$ is represented as a matrix of dimension $\mathbf{y}$-size $\times$ $\mathbf{x}$-size. For gradients with respect to matrices, their vectorized from is used. Notice,
$$\frac{\partial \mathbf{x}^l}{\partial \mathbf{W}^l} = \frac{\partial \mathbf{x}^l}{\partial \mathbf{h}^l}\frac{\partial \mathbf{h}^l}{\partial \mathbf{W}^l} = \mathbf{D}^l\frac{\partial \mathbf{h}^l}{\partial \mathbf{W}^l}\,. \tag{11}$$
Considering the feed-forward dynamics for a particular neuron i,
$$h_i^l = \sum_j W_{ij}^l x_j^{l-1} + b_i^l\,, \tag{12}$$
$$\frac{\partial h_i^l}{\partial W_{ij}^l} = x_j^{l-1}\,.$$
Therefore, using the Kronecker product, we can compactly write:
$$\frac{\partial \mathbf{x}^l}{\partial \mathbf{W}^l} = (\mathbf{D}^l)^T \otimes (\mathbf{x}^{l-1})^T\,. \tag{13}$$
Now, Eq. (10) can be written as:
$$\mathbf{g}_{\mathbf{w}^l} = (\epsilon\, \mathbf{J}^{l,K}\mathbf{D}^l)^T \otimes (\mathbf{x}^{l-1})^T\,, \tag{14}$$
$$\mathbf{g}_{\mathbf{w}^l}^T = \epsilon\, \mathbf{J}^{l,K}\mathbf{D}^l \otimes \mathbf{x}^{l-1}\,.$$
Here, $\mathbf{A}^T \otimes \mathbf{B}^T = (\mathbf{A} \otimes \mathbf{B})^T$ is used. Moreover, for linear networks $\mathbf{D}^l = \mathbf{I}$ and $\mathbf{x}^l = \mathbf{h}^l$ for all $l \in \{1 \dots K\}$. Therefore, $\mathbf{x}^{l-1}$ can be written as:
$$\begin{aligned}
\mathbf{x}^{l-1} &= \phi(\mathbf{W}^{l-1}\phi(\mathbf{W}^{l-2}\dots\phi(\mathbf{W}^1\mathbf{x}^0 + \mathbf{b}^1)\dots + \mathbf{b}^{l-2}) + \mathbf{b}^{l-1})\,, \\
&= \mathbf{W}^{l-1}(\mathbf{W}^{l-2}\dots(\mathbf{W}^1\mathbf{x}^0 + \mathbf{b}^1)\dots + \mathbf{b}^{l-2}) + \mathbf{b}^{l-1}\,, \\
&= \prod_{k=1}^{l-1}\mathbf{W}^k\mathbf{x}^0 + \prod_{k=2}^{l-1}\mathbf{W}^k\mathbf{b}^1 + \dots + \mathbf{b}^{l-1}\,, \\
&= \mathbf{J}^{0,l-1}\mathbf{x}^0 + \mathbf{a}\,,
\end{aligned} \tag{15}$$
where $\mathbf{a}$ is the constant term that does not depend on $\mathbf{x}^0$. Hence, the proof is complete. □

B Experiment settings

We evaluate for MNIST, CIFAR-10, and Tiny-ImageNet image classification tasks. We use sgd with momentum and train up to 80k (for MNIST) or 100k (for CIFAR-10 and Tiny-ImageNet) iterations. The initial learning is set 0.1 and decays by $1/10$ at every 20k (MNIST) or 25k (CIFAR-10 and Tiny-ImageNet). The mini-batch size is set to be 100, 128, and 200 for MNIST, CIFAR-10, and Tiny-ImageNet, respectively. For all experiments, we use 10% of training set for the validation set. We evaluate at every 1k and record the lowest test error. All results are the average of either 10 (for MNIST) or 5 (for CIFAR-10 and Tiny-ImageNet) runs.

When computing CS, we always use all examples in the training set to prevent stochasticity by a particular mini-batch. We also use the entire training set when computing Jacobian singular values of a network. Unless stated otherwise, we set the default pruning sparsity level to be $\bar{\kappa} = 90\%$ (*i.e.*, 90% of the entire parameters in the network is pruned away). For all tested architectures, pruning such level of sparsity does not lead to a large accuracy drop.