Stochastic Pruning for Neural Networks

Luis Alfredo Avendaño Muñoz
School of Computer Science
University of Leeds
Leeds, United Kingdom
sclaam@leeds.ac.uk

Netta Cohen†
School of Computer Science
University of Leeds
Leeds, United Kingdom
n.cohen@leeds.ac.uk

Nabi Omidvar†
School of Computer Science &
Leeds University Business School
University of Leeds
Leeds, United Kingdom
m.n.omidvar@leeds.ac.uk

***Abstract*—Pruning algorithms compress deep neural networks while incurring minimal degradation of accuracy. We propose Stochastic Pruning: an algorithm in which dense solutions are perturbed with Gaussian noise prior to pruning. We use Stochastic Pruning to explore the basin of attraction around one solution obtained with a single minimiser. Using ResNet18, ResNet50, and VGG19 on CIFAR-10 and CIFAR-100, we show that SP consistently improves one-shot accuracy after extreme pruning and mitigates feature variance explosion. Our analysis adds understanding by linking one-shot accuracy and local gradients.We show that fine-tuned SP accuracy can outperform deterministically pruned solutions in most cases. Finally, we leverage multi-objective evolutionary optimisation to compare single and two-objective results, e.g. optimised for both accuracy and sparsity. The good performance of SP provides a promising approach to extreme pruning, achieving viable high accuracy and high efficiency neural networks in resource-constrained scenarios.**

***Index Terms*—neural network pruning, stochastic pruning, sparse models, lottery ticket hypothesis.**

I. Introduction

Neural networks have achieved transformational success across domains [1], but their power is often achieved through ever larger and more complex networks, requiring ever greater computing power, memory resources, and training data, limiting accessibility, and increasing associated carbon footprints [2]. To address this, a variety of approaches have been proposed for training sparse networks [3, 4, 5], or pruning pretrained networks [6, 7, 8, 9].

Pruning methods aim to design a saliency score that enables effective network pruning [10] based on various network characteristics, such as connectivity [11, 12] and second-order information [8, 13, 14]. The simplest pruning method applied to pretrained networks is Global Magnitude Pruning (GMP) [15], which ranks all the weights globally by magnitude $|w_i|$, retains the top $(1-\gamma)$ fraction of the weights, and sets the remaining weights to zero. The pruning rate, γ, represents the fraction of weights that are removed.

Pruned networks occupy the same basin of attraction as the dense networks from which they originate [5] and exhibit robustness to minor perturbations [16]. We propose that multiple subnetworks, obtained by pruning the same densely pretrained network, can serve as a tool for investigating the local attractor. To this end, we introduce a new two-step pruning method called Stochastic Pruning (SP). First, we inject additive Gaussian noise into a single dense network to perturb it. Then, we apply pruning to the perturbed network. We refer to the resulting pruned networks as stochastic models. Our findings indicate that, across varying datasets and models, the application of SP with high pruning rates yields superior one-shot performance compared to deterministically pruned networks (applied to the original, unperturbed network).

This is relevant since there is an increasing interest in using deep learning in IoT devices, where fine-tuning/training is costly or simply not possible [17]. Alternatively, one-shot pruning may suffice in cases where it is competitive with iterative pruning, e.g. as observed in some cases for CNNs [16]. Finally, one-short pruning is useful as a first step before fine tuning, since in our experiments, the advantage of stochastic pruning is preserved after fine-tuning, albeit dependent upon the pruning methodology. For more sophisticated pruning methods, it is evident that the noise amplitude exhibits an inverse relationship with the final fine-tuned performance, as it affects the gradient flow, thus reducing the efficacy of training [18].

Our procedure involves two adjustable parameters: the noise amplitude σ and the pruning rate γ. To systematically search for 'good' parameters, we used two approaches: an exhaustive grid-search, quantifying one objective at a time, and multi-objective evolutionary optimisation. Next, we identify a phenomenon we call feature variance explosion, in which highly (deterministically) pruned networks have a much greater variance in their feature maps compared to their dense counterpart. The feature variance interferes with the inference process and hence the accuracy. We find that SP alleviates this explosion, explaining in part how stochastic pruning enhances one-shot performance.

Our main contributions are:

- We show that stochastic pruning of pre-trained networks (SP) outperforms deterministic GMP in one-shot settings, with fine-tuned models often matching or exceeding deterministic ones.
- We demonstrate that the improved performance of SP solutions arises from the combination of (i) the pruning mask and (ii) altered weights.
- We show that using SP with GMP one-shot SP generally alleviates *the feature variance explosion*.

† Authors had equal contributions

- We demonstrate a trade-off between one-shot accuracy and trainability of SP solutions mediated by the noise level σ

The remainder of this paper is organised as follows. After introducing related work (Section II), the methodology of Stochastic Pruning is described (Section III). We evaluated the different contributing factors to the performance of stochastic solutions (IV-A), and demonstrated the generalisation of our results to different models and test data (IV-B). In IV-C, we use multi-objective optimisation to extend our analysis. Mechanistic insight in terms of feature variance explosion is presented in IV-D. Finally, in IV-E, we show that stochastic models provide good solutions after fine-tuning. The paper ends with a brief conclusion (Section V).

II. Related Work

Approaches to compressing neural networks vary, ranging from designing efficient architectures from scratch [19] to pruning-based methods. These include pruning at initialisation [11, 12, 20] and dynamical sparse training, where the model's sparse connections evolve during training [3, 21]. Among these techniques, pruning remains the most widely used due to its simplicity and practicality. In general, pruning algorithms can be classified into three categories: *pre-training*, *training*, and *post-training* pruning.

Pre-Training Pruning: This family of algorithms prunes a network on randomly initialised weights, i.e. before training. Synaptic flow (SynFlow) [11] does not use any data for pruning and yet is demonstrated to avoid layer collapse. Gradient Signal Preservation (GraSP) [20] prune randomly initialised networks while preserving the gradient flow signal throughout the network. Yet a different approach, Single-Shot Network Pruning Based on Connection Sensitivity [12] adds parameters that represent connections (but not weights) between neurons and uses the approximate gradient of the loss function with respect to the new parameters as a salience score for pruning.

The Lottery Ticket Hypothesis (LTH) [16] states that there exists a sparse subnetwork (winning ticket) within a dense model that, when trained in isolation, can achieve comparable or better performance than the original dense network. To unveil a winning ticket it is necessary to sequentially perform Global Magnitude Pruning (GMP) [15] on a trained network, transfer the final mask on the initial weights, and train this sparse model from scratch. These sparse models, obtained through iterative version of GMP, where pruning is repeated across multiple cycles to gradually reach the desired sparsity [22], are robust to small perturbations that do not change this basin [18]. For example, using only the weight sign preserves LTH performance, while shuffling weights does not, with the latter transporting solutions to a different basin [23].

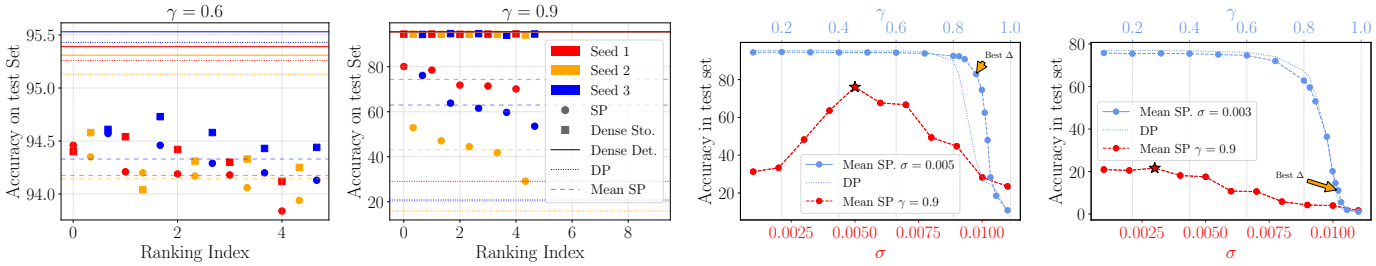
Training: These algorithms prune connections during the training process, some prune gradually until reaching the final sparsity [16]. This approach, commonly known as Iterative Magnitude Pruning (IMP), keeps the mask fixed after pruning. In contrast to IMP, Dynamical Sparse Training (DST) methods dynamically change the mask during training, starting with a sparse model before training, then pruning and regrowing the remaining connections based on different criteria. The authors in [3] randomly prune a portion of the surviving connections and then regrow them following an Erdös-Rényi distribution. Sparse Networks from Scratch (SNFS) [4] regrows weights based on the momentum of each weight, reallocating the weights from the less efficient layers to more weight-efficient layers. Authors in [21] prune a fraction of the surviving weights randomly but regrow the weights based on the gradients of the zero connections, reactivating the connection that would incur the maximum loss decrease.

Post-Training pruning: These techniques perform pruning after the training phase and are typically accompanied by a fine-tuning schedule. Each method has a different importance score to detect unimportant weights. The simplest is the magnitude of a weight [24] where the lowest-magnitude weights are deemed unnecessary. However, this method can remove weak but important weights. Some methods use first-order Taylor expansion coefficients of the loss function (i.e. gradient information) as an importance score for pruning [6, 7]. Hessian-based methods measure the importance of weights as the effect they have in the second-order approximation of the training loss function [8, 9]. Many of these methods require a fine-tuning phase to mitigate the accuracy drop, especially for high pruning rates.

In this study, we focus on post-training pruning. This approach capitalises on the availability of pretrained models for a multitude of tasks, while acknowledging that the network size often poses a limitation for deployment in real-world scenarios, e.g. off the network, in mobile phones, and on edge devices. Our experimental analysis predominantly used Global Magnitude Pruning (GMP); however, we also compared a subset of our findings against the layer adaptive magnitude pruning method (LAMP) [10]. In contrast to GMP, LAMP computes importance scores for each weight within its respective layer. These scores quantify the relative significance of each connection, ensuring that the smaller magnitude weights within a layer are pruned first. The computed scores are then aggregated and sorted globally across the entire network, followed by global magnitude pruning. Consequently, weights with identical magnitudes can receive different LAMP scores based on their layer context and relative importance. This approach enables magnitude pruning to achieve strong performance without requiring manually defined layer-specific sparsity levels. Magnitude pruning can generally yield surprisingly effective results when properly combined with layer-specific pruning rates [21, 25].

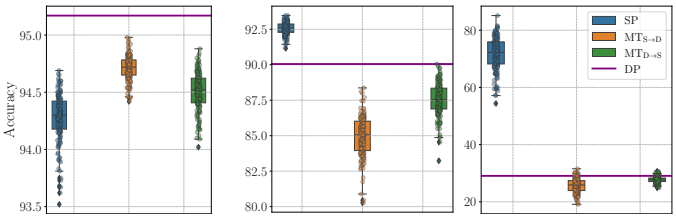
III. Stochastic Pruning

Neural network pruning relies on the finding that networks are often overparameterised, allowing for similar performance with a smaller network [15]. Different pruning approaches aim to find a sparse network that maintains the performance of the original, or at least degrades gracefully. Following Lee *et al.* [12] we formulate the problem as follows: Given a dataset $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^{n}$ and a desired pruning rate γ (defined as



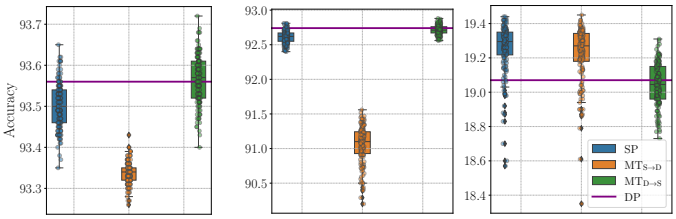
(a) One-shot performance of different models (CIFAR10, 5 models, 3 seeds) (b) CIFAR10 (c) CIFAR100

Fig. 1: **Unveiling good One-Shot sparse solutions:** Using ResNet18 on CIFAR10/100 for: **a)** CIFAR10 test accuracy, comparing three ResNet18 seeds with SP parameters $\sigma = 0.005$ and $\gamma = \{0.6, 0.9\}$. For each seed, one deterministic and a population of five stochastic solutions were generated. **b)** and **c)** show test set accuracy as a function of γ and σ, exploring values around optimal parameters (see Table I). The arrow indicates the pruning rate yielding the best difference between SP and Deterministic Pruning (DP). The star indicates the noise level yielding the best performance.



(a) $\gamma = 0.5$ (b) $\gamma = 0.8$ (c) $\gamma = 0.9$

Fig. 2: **Mask transfer ResNet18 on CIFAR10:** SP one-shot performance draws from both the mask and stochastic weights, showing: SP (blue, left), $\text{MT}_{\text{S}\to\text{D}}$ (orange, middle) and $\text{MT}_{\text{D}\to\text{S}}$ (green, right). $\sigma = 0.005$. purple line: deterministic pruning.



(a) $\gamma = 0.72$ (b) $\gamma = 0.88$ (c) $\gamma = 0.94$

Fig. 3: **Mask transfer VGG19 on CIFAR10:** SP one-shot performance generalises to different models, showing: SP (blue, left), $\text{MT}_{\text{S}\to\text{D}}$ (orange, middle) and $\text{MT}_{\text{D}\to\text{S}}$ (green, right). $\sigma = 0.001$. purple line: deterministic pruning.

the fraction of total weights that are zero), the optimisation problem for neural network pruning can be stated as follows:

$$\begin{aligned} \min_{\mathbf{w},\mathbf{m}} \quad & \frac{1}{n}\sum_{i=1}^{n} \ell(F(\mathbf{m}\odot\mathbf{w}; x_i), y_i) \\ s.t. \quad & \mathbf{w} \in \mathbb{R}^d, \\ & \mathbf{m} \in \{0,1\}^d \quad \|\mathbf{m}\|_0 \geq \lfloor (1-\gamma)\cdot d \rfloor, \end{aligned} \tag{1}$$

where $\ell(\cdot)$ is the loss function, $\odot$ is the Hadamard product, $F(\mathbf{m}\odot\mathbf{w}; x)$ is the output of the neural network given the set of weights $\mathbf{w}$ and masks $\mathbf{m}$, d is the total number of parameters, and $\|\cdot\|_0$ is the ℓ_0-norm. Most post-training pruning methods reveal a static mask $\mathbf{m}$ and fine-tune the remaining weights $\tilde{\mathbf{w}} = \mathbf{m}\odot\mathbf{w}$ to minimise the accuracy drop. Stochastic pruning is represented by the following equations:

$$\mathbf{w}_\text{p} = \mathbf{w}^* + \mathcal{N}(\mathbf{0}, \sigma^2\mathbf{I}) \tag{2}$$

$$\hat{\mathbf{w}} = \mathbf{m}(\mathbf{w}_\text{p}) \odot \mathbf{w}_\text{p}, \tag{3}$$

where $\mathbf{w}^*$ is a pre-trained set of weights, $\mathcal{N}(\mathbf{0}, \sigma^2\mathbf{I})$ represents Gaussian noise with standard deviation σ, and $\mathbf{m}(\mathbf{w}_\text{p})$ is the mask dependent on the perturbed weights $\mathbf{w}_p$ via some function $f(\cdot)$. For GMP, $f(\cdot) = |\cdot|$, and for LAMP, $f(\cdot)$ corresponds to the LAMP score [10]. Equation (2) produces an intermediate network, which we call "Dense Stochastic". Finally, we call the original pruned network with $\sigma = 0$, "deterministic".

IV. Experiments

We evaluated our Stochastic Pruning approach using three models – ResNet18, ResNet50, and VGG19 – and two datasets, CIFAR-10 and CIFAR-100. All models were adapted for the CIFAR datasets*. The pretrained models were trained for 200 epochs using Stochastic Gradient Descent (SGD) with an initial learning rate of 0.1, a cosine annealing schedule ($T_\text{max} = 200$), and a batch size of 128.

For post-pruning fine-tuning experiments, we used SGD with 100 epochs, an initial learning rate of 0.0001, a cosine annealing schedule ($T_\text{max} = 100$), a weight decay of 5×10^{-5} and a gradient clipping of 0.1.

In Section IV-A, we analyse the one-shot performance of stochastic pruning compared to their deterministic counterparts for various γ values. To clarify the sources of its superior performance, we examine the effect of σ and γ on stochastic pruning (SP), conducting a mask transfer analysis. In Section IV-B, we perform a grid search over SP's hyperparameters, σ and γ, which illustrates that stochastic pruning consistently outperforms deterministic pruning when optimal hyperparameter values are used. In Section IV-C, we use a multi-objective evolutionary optimisation framework to explore the trade-offs between stochastic performance and γ,

*Models adapted from https://github.com/kuangliu/pytorch-cifar

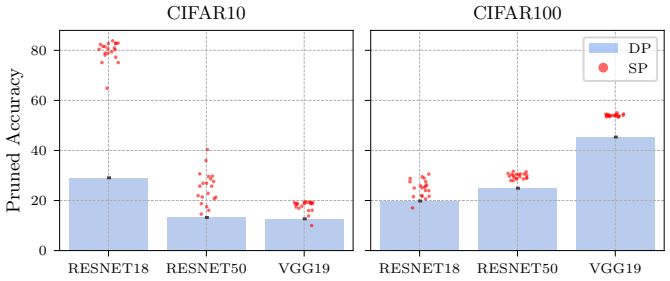


Fig. 4: SP one-shot performance using the optimal Grid Search parameters in Table I.

TABLE I: Optimal parameters for one-shot SP for Grid and MOO search, all SP measures are the median of 5 models

Dataset	Model	Grid Search					MOO Search - F_2				
		γ	σ	SP	DP	Δ	γ	σ	SP	DP	Δ
CIFAR10	ResNet18	0.9	0.005	80.9	29.0	**51.9**	0.87	0.0038	88.4	39.7	**48.6**
	ResNet50	0.95	0.003	29.3	13.2	**13.1**	0.94	0.0028	33.9	16.3	**17.6**
	VGG19	0.95	0.003	19.3	12.7	**6.6**	0.91	0.0013	54.6	40.8	**13.7**
CIFAR100	ResNet18	0.9	0.003	26.3	19.7	**6.5**	0.92	0.0036	13.9	5.18	**8.7**
	ResNet50	0.85	0.001	30.5	24.9	**5.6**	0.76	0.0012	63.2	59.9	**3.3**
	VGG19	0.8	0.001	53.7	45.3	**8.4**	0.83	0.0025	28.4	16.5	**11.9**

on the one hand, and between performance and performance gain (over deterministic models) on the other.

A. *Good One-Shot Sparse Solutions*

There is both theoretical [26] and empirical [27] evidence highlight the vital role of noise during the training of neural networks. Furthermore, it is known that LTH solutions are recoverable when training has become stable to SGD noise [22]. Hence, we can explore the local basin of a pretrained network by injecting Gaussian noise, with the knowledge that this does not affect the probability of finding a good sparse solution (LTH) as by the late epochs of training, the network has already become stable to SGD noise. Nonetheless, one would naively expect that randomly injecting noise to a highly trained network should not provide a performance advantage. Surprisingly, we found that for high pruning rates and appropriately chosen noise levels, SP generates much better solutions than their deterministically pruned counterparts.

In Figure 1a different ResNet18 seeds were trained in CIFAR10 for different pruning rates. All stochastic dense networks maintain similar performance to the original dense network.

For an intermediate pruning rate ($\gamma = 0.6$), individual stochastic models can sometimes achieve higher accuracy, but on average, SP performance is slightly worse than deterministic pruning, although, the accuracy drop is similar ($\approx 1\%$) for both stochastic and deterministically pruned models. As accuracy drops with higher pruning rates, SP begins to outperform deterministic models. At $\gamma = 0.9$, SP is by far superior to deterministic pruning (with rare exceptions, not shown). This advantage is only unveiled when pruning is performed, since dense stochastic networks (coloured squares) perform similarly to the dense original (unperturbed pretrained) models. The consistency of the results and in particular the crossover of SP performance around the transition to high pruning rates (and degraded accuracy) appears to indicate a general principle underpinning the success of SP. As the robustness of our experiments demonstrates that SP is not confined to one seed, seed 1 was used for the results in the remainder of the paper.

In Figures 1b and 1c, SP performance is presented as a function of σ and γ for the CIFAR10 and CIFAR100 test sets, respectively. For the CIFAR10 dataset, a discernible optimal noise level emerges around $\sigma = 0.005$. Above the crossover point ($\gamma \approx 0.8$), SP significantly outperforms the deterministic approach. This phenomenon is also observed with the CIFAR100 dataset, although the peak performance for σ and the disparity between SP and deterministic performance (DP) are diminished, presumably because this disparity occurs for pruning rates with a considerably degraded performance, relative to the dense network. This can be attributed to CIFAR100 being a more difficult task.

As SP is a two-step algorithm, the source of the improved one-shot accuracy is ambiguous. Each of the steps suggests a possible factor: the perturbation of the weights themselves, and the stochastically generated mask. We consider these in turn, asking whether stochastic weights $\mathbf{w}_\mathrm{p}$, stochastic mask $\mathbf{m}(\mathbf{w}_\mathrm{p})$, or a combination of both $\hat{\mathbf{w}} = \mathbf{m}(\mathbf{w}_\mathrm{p}) \odot \mathbf{w}_\mathrm{p}$ are responsible for enhanced performance. To this end, we define two operations:

- $\mathrm{MT}_{\mathrm{S}\to\mathrm{D}}$: Apply the pruning mask from stochastic models to the original deterministic weights, i.e. $\mathbf{w}^* \odot \mathbf{m}(\mathbf{w}_\mathrm{p})$.
- $\mathrm{MT}_{\mathrm{D}\to\mathrm{S}}$: Apply the pruning mask from the deterministic model to the dense stochastic model, i.e. $\mathbf{m}(\mathbf{w}^*) \odot \mathbf{w}_\mathrm{p}$.

If operation $\mathrm{MT}_{\mathrm{S}\to\mathrm{D}}$ results in superior models relative to deterministic pruning, it would indicate that stochastic pruning has the potential to uncover effective masks. In contrast, if operation $\mathrm{MT}_{\mathrm{D}\to\mathrm{S}}$ results in models superior to deterministic pruning, it indicates that the weights identified through stochastic pruning are superior to the deterministic weights in the pruned networks.

For this experiment, we obtained 80 stochastically pruned models for ResNet18 on CIFAR10 ($\sigma = 0.005$, Figure 2). To test whether the architecture changes the results of stochastic pruning, or the contribution of the mask and weights, we obtained 80 SP solutions using VGG19, also tested on CIFAR10 ($\sigma = 0.001$, Figure 3). The pruning rates for VGG19 were selected such that both models have a similar number of non-zero weights.

Our results demonstrate that the high accuracy achieved by SP generalises across architectures, but that the relative contributions of the two operations varies. For ResNet18, stochastic weights and the stochastic masks (blue box plot) are both required to explain SP's accuracy. Neither operation alone outperforms deterministic pruning, but their combination yields the best accuracy (outperforming deterministic pruning for sufficiently high pruning rates). This phenomenon is particularly strong for extreme pruning rates ($\gamma = 0.9$). For VGG19, transferring the stochastic mask to the original unperturbed weights yields similar performance to deterministic pruning

TABLE II: **Accuracy and GF for one shot and fine-tuned:** Parameters as Table I

Dataset	Model	LAMP								GMP								GRASP							
		One-Shot					Fine-Tuned			One-Shot					Fine-Tuned			One-Shot					Fine-Tuned		
		SP		DP		Δ	SP	DP	Δ	SP		DP		Δ	SP	DP	Δ	SP		DP		Δ	SP	DP	Δ
		GF	Acc	GF	Acc					GF	Acc	GF	Acc					GF	Acc	GF	Acc				
CIFAR10	ResNet18	9.9	84.1	18.9	77.8	**6.3**	91.9	92.8	-0.9	8.0	80.9	11.5	29.0	**51.8**	91.8	89.6	**2.2**	$4.6{\cdot}10^3$	10.0	11.9	10.0	0.0	43.9	47.0	-3.1
	ResNets50	35.9	28.4	89.6	22.4	**6.0**	88.8	91.2	-2.4	16.0	29.3	19.1	13.2	**16.1**	90.6	90.6	0.0	12.0	10.0	37.9	10.0	0.0	24.8	25.5	-0.7
	VGG19	62.9	20.4	8.7	88.8	-68.4	91.1	92.6	-1.5	46.0	19.3	46.2	12.7	**6.6**	92.7	92.3	**0.4**	5.8	10.0	3.9	10.0	0.0	10.1	10.0	**0.1**
CIFAR100	ResNet18	54.4	26.1	100.1	18.3	**7.8**	66.8	67.6	-0.8	18.4	26.3	27.0	19.7	**6.6**	67.6	66.9	**0.7**	$2.1{\cdot}10^4$	1.0	$1.3{\cdot}10^5$	1.0	0.0	10.3	10.0	**0.3**
	ResNets50	23.4	38.7	29.5	30.0	**8.7**	72.4	72.1	**0.3**	26.6	30.5	27.1	24.9	**5.6**	74.1	73.9	**0.2**	$1.8{\cdot}10^4$	1.0	$7.4{\cdot}10^4$	1.0	0.0	1.1	1.6	-0.5
	VGG19	6.6	69.3	7.3	68.8	**0.5**	71.9	72.0	-0.1	18.4	53.7	23.3	45.3	**8.4**	72.4	72.2	**0.2**	307.0	1.0	766.7	1.0	0.0	6.0	3.2	**2.9**

TABLE III: **Feature Variance explosion:** Parameters as in Table I.

Model	One-Shot				Fine-tuned			
	CIFAR10		CIFAR100		CIFAR10		CIFAR100	
	DP	SP	DP	SP	DP	SP	DP	SP
ResNet18	0.759	**0.698**	0.720	**0.675**	**0.999**	1.003	**1.000**	1.001
ResNet50	0.423	**0.401**	0.479	**0.477**	1.000	**0.972**	1.000	**0.999**
VGG19	**0.467**	0.470	0.739	**0.736**	1.000	**0.990**	**0.999**	1.000

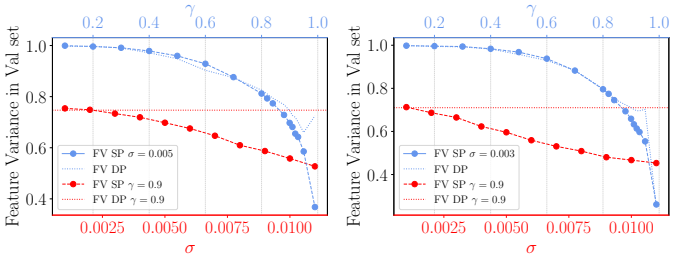
for all three pruning rates tested, but the combined SP outperforms deterministic pruning only for extreme pruning rates ($\gamma = 0.94$), where the stochastic weights play a key role. It is important to note that the optimal noise levels (necessary to achieve good performance) differ across experiments, depending on both the model and the dataset.

In summary, SP reveals effective sparse models that surpass deterministic pruning. We found examples in which this performance arises from the combination of stochastic weights and the stochastically derived mask (ResNet18), or where the mask itself is sufficient to produce the upgraded performance (VGG19). Further experiments would be needed to explore the mechanistic reasons underpinning these differences.

B. Generalisability of Stochastic Pruning

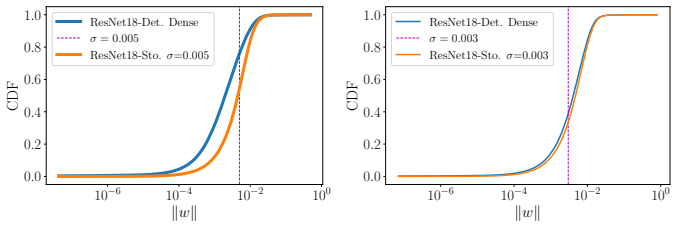
We have already seen that SP provides promising solutions in the extreme pruning rate regime, for both ResNet18 and VGG19. Here, we ask whether and to what extent does SP performance surpass deterministic pruning across 3 models (adding ResNet50) and the two datasets (CIFAR10, CIFAR100). A restricted grid search for the pruning rate was conducted with $\gamma = \{0.8, 0.9, 0.95\}$ and $\sigma = \{0.001, 0.003, 0.005\}$ for each model-dataset combination (the best parameters are in Table I). To this end, we used the median one-shot performance of ten SP models, and found that for all models and datasets and across the three (extreme) pruning rates tested, SP models are statistically superior to deterministic ones (all with p-values< 0.05).

To ensure that this result applies robustly, we consider individual solutions (20 stochastically pruned models per model and dataset pair). We find that, with only rare exceptions, SP solutions consistently demonstrate superior performance relative to deterministic pruning (Figure 4).



(a) CIFAR10 (b) CIFAR100

Fig. 5: **Feature Variance Across σ and γ:** ResNet18 on CIFAR10/100.



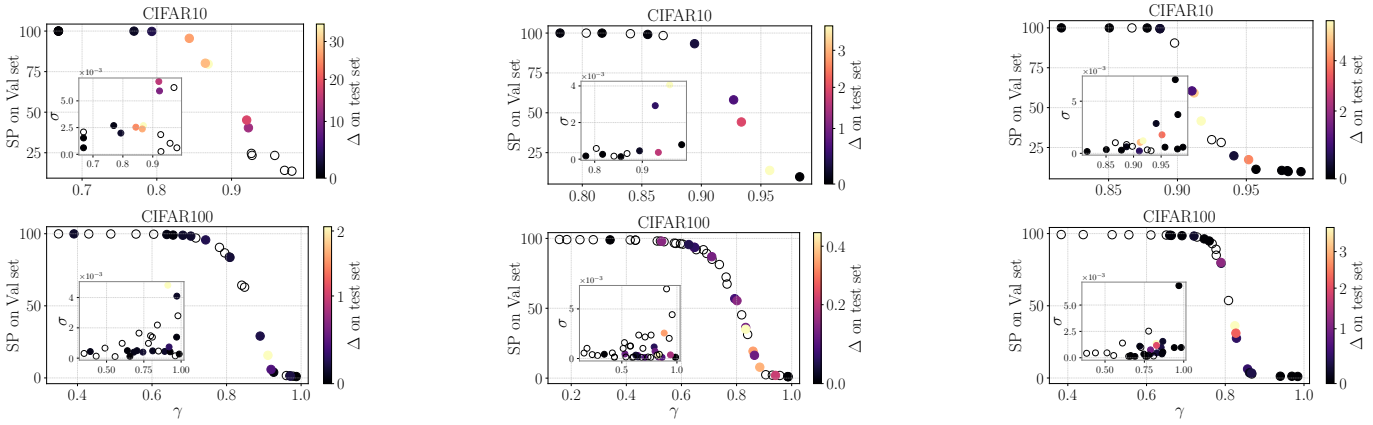
(a) CIFAR10 (b) CIFAR100

Fig. 6: Weight's CDF for ResNet18 trained on CIFAR10/100.

C. Exploration of hyperparameter space with multi-objective evolutionary optimisation

To more thoroughly explore the hyperparameter space, to more efficiently search for optimal hyperparameters, and to expand our search to additional objectives, we used a Multi-objective Evolutionary algorithm, namely, NSGA-II [28].

NSGA-II is a genetic algorithm for multi-objective optimisation that maintains a population of solutions, using non-dominated sorting and crowding distance to select high-performing, diverse individuals. After a set number of generations, it outputs the non-dominated solutions (Pareto front) found in the final population. We used two sets of functions consisting of two fitness functions ($F_1 = \{f_{11}, f_{12}\}, F_2 = \{f_{21}, f_{22}\}$). In the initial search (using F_1), we sought solutions that maximise performance and pruning rates, setting our objective function at $F_1 = \{f_{11}, f_{12}\} = \{\text{Acc}_{\text{validation}}(\hat{\mathbf{w}}), \gamma\}$, where $\text{Acc}_{\text{validation}}(\hat{\mathbf{w}})$ denotes the median performance of 10 stochastic models in the validation set. Here, γ serves both as a decision variable and as an objective. This approach seeks to



(a) ResNet18 (b) ResNet50 (c) VGG19

Fig. 7: **Pareto Front $\mathbf{F_1}$:** Median SP performance ($n = 10$) versus γ in the validation set for F_1. Δ is calculated on the test set. Every combination has a pruning rate region in which SP surpasses deterministic pruning, especially in the transition region (to high pruning rates). Bright colours indicate the greatest improvement. Empty circles indicate solutions with better DP solutions.

verify whether there exists a range of pruning rates that allows SP to outperform DP without directly optimising for it. In the second search, we use the set of fitness functions defined by $F_2 = \{f_{21}, f_{22}\} = \{\text{Acc}(\hat{\mathbf{w}}), (\text{Acc}(\hat{\mathbf{w}}) - \text{Acc}(\mathbf{w})) \cdot \gamma\}$, where the two objectives are to maximise the median performance of SP and to maximise the disparity between stochastic and deterministic pruning ($\text{Acc}(\mathbf{w})$). We multiply the difference by the pruning rate to encourage solutions that demonstrate a pronounced difference between stochastic and deterministic pruning at elevated pruning rates.

Accuracies at search time were calculated using 5000 training images in all cases. The search space for both functions sets is $\sigma \in (0.0001, 0.01)$ and $\gamma \in (0.01, 0.99)$. We sampled 150 individuals in the hyperparameter space for both functions sets and we obtained the Pareto Front form them. The combination ResNet18-CIFAR10 produces the highest improvement over DP in the test set, while all others have modest improvements, with maximum improvements between 0.4% and 5% (Figure 7). We note the S-shaped Pareto fronts. These are to be expected, since large pruning rates invariably worsen one-shot performance. A more intriguing aspect of these figures is that the most substantial improvement over deterministic pruning in the test set is observed at high or extreme γ, between 85-94% for CIFAR10 and 75-95% for CIFAR100. While we found a wide range of pruning rates in which stochastic pruning can outperform deterministic pruning, this range is mostly confined to the pruning rates where performance starts to rapidly deteriorate, but does not render the network useless yet.

In Table I are the results for the parameters that are present in the Pareto front for F_2 and maximise f_{22}. Values are calculated over the test set. In particular, SP outperforms deterministic pruning in the high γ and low σ regimes, indicating that NSGA-II effectively identify hyperparameters where SP excels.

D. Feature Variance Explosion

Why does a simple procedure, such as introducing noise followed by pruning, substantially improve one-shot performance? Introducing noise to the weights increases the variance of the weight distribution, as $\text{Var}(\mathbf{w} + e) = \text{Var}(\mathbf{w}) + \text{Var}(e)$ due to e being independent of $\mathbf{w}$, resulting in a bimodal distribution with more extreme weights after pruning. This distribution potentially enhances model accuracy by structuring pruned filters more effectively. This, we hypothesise, mitigates the *feature variance explosion*, characterised by uneven scaling of feature maps throughout the network.

First, we measure the variance of each neuron across a batch of training data. Then, we sum all the neurons variances across the layer. Finally, if we let this sum of the j^{th} layer be denoted ϕ_j for pruned models (either stochastic or deterministic) and Φ_j for dense models, we return the ratio $v_j = \frac{\phi_j}{\Phi_j}$ for each layer and we calculate the model's feature variance as follows:

$$V = \frac{1}{N\,L} \sum_{i}^{N} \sum_{j}^{L} v_j(x_i), \tag{4}$$

where N is the number of batches and L is the number of layers in the network. In the case of stochastic models, we calculated $\phi_j(x_i)$ for five different stochastic models, then averaged these results to obtain a single $\bar{\phi}_j$. We divided this average by the variance of the dense stochastic model, Φ_j, for a particular layer. We found that for most combinations, the stochastically pruned model has a lower value of V (see Table III), suggesting that alleviating feature variance explosion is a putative mechanism for enhancing one-shot performance in SP. When we fine-tune these models, we see nearly identical values of V across the board, which is expected since the internal dynamics of the pruned network resemble those of a trained network. In addition, in Figure 5, we can see that SP mainly alleviates the feature variance explosion for high pruning rates (≥ 0.8), that is, a regime in which the extreme

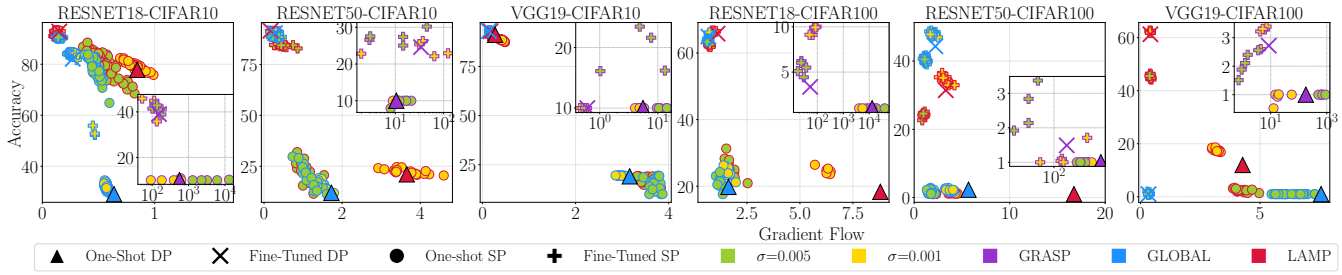


Fig. 8: **Fine-tuned Accuracy and Gradient Flow for all datasets and architectures:** We used $\gamma = \{0.9, 0.95, 0.94\}$ for ResNet18, ResNet50 and VGG19 respectively to ensure a similar number of parameters between these architectures. On average, increased performance suppresses gradient flow. High-σ models outperform DP, but low-σ models are better after fine-tuning.

sparsity of the network due to pruning results in a performance degradation. In this regime, the surviving weights matter more, and the effect of injecting noise (to weights and/or to the mask) is expected to be highly deleterious. Indeed, in fig. 6 the Cumulative Density Function (CDF) shows that for a $\sigma = 0.005$ more than half of the network's weights are smaller than the noise amplitude, suggesting that a substantial portion of the model is overshadowed by the noise. It is therefore no surprise that a large pruning rate is required to see an effect on feature variance.

E. Fine-Tuning Stochastic Solutions

So far, we have explored the one-shot regime of stochastically pruned models. In this section, we explore the robustness of fine-tuned solutions to stochastic pruning. To this end, we analyse the behaviour of stochastic GMP and a closely related but more subtle pruning method, LAMP [10]. LAMP is selected for comparison because of its status as a more sophisticated pruning method that nevertheless employs the weight magnitude to compute the saliency score, thereby maintaining computational efficiency. Our analysis focusses on the trade-offs between trainability and performance (Table II and Fig. 8), using gradient flow (GF) – a critical metric of trainability for sparse models [18], often used as a proxy of their trainability. We find that fine-tuned SP does not perform well with LAMP.

Why does SP with LAMP not perform as well after fine-tuning? One possibility is that the bar is too high to cross, namely, because DP for LAMP achieves remarkably high performance compared to DP with GMP, leaving little room for improvement. The high bar, combined with the relatively poor one-shot performance, is therefore insufficient to outperform DP. We also note that fine-tuned SP with GMP tends to perform better than fine-tuned SP models with LAMP for larger models (Table II). Once again, we attribute this to a combination of one-shot performance and the effect of fine tuning.

For GMP, a one-shot boost in performance can help the stochastic model outperform the deterministic model despite presenting a smaller GF (ResNet18-CIFAR10 in Figure 8). In contrast, for LAMP (with $\sigma = 0.005$), the initial increase in accuracy is insufficient for fine-tuned models to outperform the deterministic model.

In general, Figure 8 reveals a similar inverse relationship for both LAMP and GMP: One-shot models with higher one-shot accuracies tend to have lower GF, suppressing trainability during fine-tuning, and hence losing their advantage over DP, and even risking lower performance. The high level of noise in these models ($\sigma = 0.005$) is therefore detrimental to finding competitive fine-tuned solutions. However, models generated with more modest noise levels ($\sigma = 0.001$) maintain their advantage and can even outperform DP models after fine-tuning.

One question that arises from these observations is whether, given a set level of accuracy, maximising gradient flow is always beneficial. To test this hypothesis, we used GraSP [20], which explicitly preserves gradient flow during pruning. Using GraSP, the GFs obtained are orders of magnitude larger, up to $O(10^4)$ (Figure 8 and Table II). Not only do such high GF fail to enhance trainability, but in fact, they suppress it. In our results, optimising GF is only fruitful for GF< 10. We note that as GraSP was designed to operate at initialisation (before training), its unbounded maximisation of GF improves trainability in random networks, but not in pre-trained ones.

V. Conclusion and Discussion

We introduced stochastic pruning, both as a tool to explore basins of attraction and as a pruning algorithm that is cost-effective and simple to implement which is ideal for resource constrained environments. We found that SP improves over deterministic pruning by enabling exploration within the basin of attraction. Given that surviving weights are robust to small perturbations [18], SP effectively causes weight juxtaposition near the pruning threshold, leading to a reconfiguration of the remaining network. Furthermore, since solutions tend to converge to the periphery of local optima rather than their exact centres [29], these perturbations can further facilitate beneficial shifts in the sparse subnetwork. The perturbation-induced adjustments in SP can therefore enhance performance, akin to the second-order corrections in Optimal Brain Surgeon [14], suggesting that stochasticity may provide a more efficient mechanism for discovering better sparse subnetworks.

Our analysis shed light on the tradeoffs of different pruning rates, noise levels and pruning algorithms, in the one-shot and fine-tuning settings. We found that feature variance explosion is a concern for post-training pruning. Previously, [30]

demonstrated that adjusting the running statistics of the batch normalisation layer using a small subset of data significantly improves one-shot performance, which is strongly correlated with final fine-tuning accuracies. Combined with our results, this suggests a fundamental importance for understanding feature variance and rein it in during pruning.

ACKNOWLEDGMENTS

The authors would like to thank Hao Tan for his valuable insights and contributions to this work. Netta Cohen acknowledges funding from EPSRC (EP/S016813/1) and BBSRC (BB/Z514317/1).

REFERENCES

[1] T. Brown *et al.*, "Language Models are Few-Shot Learners," in *Advances in Neural Information Processing Systems*, vol. 33. Curran Associates, Inc., pp. 1877–1901. [Online]. Available: https://papers.nips.cc/paper/2020/hash/1457c0d6bfcb4967418bfb8ac142f64a-Abstract.html

[2] N. Thompson, K. Greenewald, K. Lee, and G. F. Manso, "The Computational Limits of Deep Learning," in *Computing within Limits*. LIMITS. [Online]. Available: https://limits.pubpub.org/pub/wm11wjce/release/1

[3] D. C. Mocanu, E. Mocanu, P. Stone, P. H. Nguyen, M. Gibescu, and A. Liotta, "Scalable training of artificial neural networks with adaptive sparse connectivity inspired by network science," vol. 9, no. 1, p. 2383. [Online]. Available: https://www.nature.com/articles/s41467-018-04316-3

[4] T. Dettmers and L. Zettlemoyer, "Sparse Networks from Scratch: Faster Training without Losing Performance." [Online]. Available: http://arxiv.org/abs/1907.04840

[5] U. Evci, F. Pedregosa, A. Gomez, and E. Elsen, "The Difficulty of Training Sparse Neural Networks," in *ICML 2019 Workshop on Identifying and Understanding Deep Learning Phenomena*, May 2019.

[6] E. Karnin, "A simple procedure for pruning back-propagation trained neural networks," vol. 1, no. 2, pp. 239–242.

[7] M. C. Mozer and P. Smolensky, "Skeletonization: A Technique for Trimming the Fat from a Network via Relevance Assessment," in *Advances in Neural Information Processing Systems*, vol. 1. Morgan-Kaufmann. [Online]. Available: https://proceedings.neurips.cc/paper/1988/hash/07e1cd7dca89a1678042477183b7ac3f-Abstract.html

[8] Y. LeCun, J. Denker, and S. Solla, "Optimal Brain Damage," in *Advances in Neural Information Processing Systems*, vol. 2. Morgan-Kaufmann. [Online]. Available: https://proceedings.neurips.cc/paper/1989/hash/6c9882bbac1c7093bd25041881277658-Abstract.html

[9] S. P. Singh and D. Alistarh, "WoodFisher: Efficient Second-Order Approximation for Neural Network Compression," in *Advances in Neural Information Processing Systems*, vol. 33. Curran Associates, Inc., pp. 18 098–18 109. [Online]. Available: https://papers.nips.cc/paper/2020/hash/d1ff1ec86b62cd5f3903ff19c3a326b2-Abstract.html

[10] J. Lee, S. Park, S. Mo, S. Ahn, and J. Shin, "Layer-adaptive Sparsity for the Magnitude-based Pruning," in *International Conference on Learning Representations*. [Online]. Available: https://openreview.net/forum?id=H6ATjJ0TKdf

[11] H. Tanaka, D. Kunin, D. L. Yamins, and S. Ganguli, "Pruning neural networks without any data by iteratively conserving synaptic flow," in *Advances in Neural Information Processing Systems*, H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, Eds., vol. 33. Curran Associates, Inc., pp. 6377–6389. [Online]. Available: https://proceedings.neurips.cc/paper/2020/file/46a4378f835dc8040c8057beb6a2da52-Paper.pdf

[12] N. Lee, T. Ajanthan, and P. Torr, "SNIP:Single-Shot Network Pruning Based on Connection Sensitivity," in *International Conference on Learning Representations*, Sep. 2018.

[13] B. Hassibi and D. Stork, "Second order derivatives for network pruning: Optimal Brain Surgeon," in *Advances in Neural Information Processing Systems*, vol. 5. Morgan-Kaufmann. [Online]. Available: https://proceedings.neurips.cc/paper/1992/hash/303ed4c69846ab36c2904d3ba8573050-Abstract.html

[14] B. Hassibi, D. Stork, and G. Wolff, "Optimal Brain Surgeon: Extensions and performance comparisons," in *Advances in Neural Information Processing Systems*, vol. 6. Morgan-Kaufmann. [Online]. Available: https://proceedings.neurips.cc/paper/1993/hash/b056eb1587586b71e2da9acfe4fbd19e-Abstract.html

[15] M. Gupta, E. Camci, V. R. Keneta, A. Vaidyanathan, R. Kanodia, A. James, C.-S. Foo, M. Wu, and J. Lin, "Is Complexity Required for Neural Network Pruning? A Case Study on Global Magnitude Pruning," in *2024 IEEE Conference on Artificial Intelligence (CAI)*, pp. 747–754. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/10605398

[16] J. Frankle and M. Carbin, "The Lottery Ticket Hypothesis: Finding Sparse, Trainable Neural Networks," in *International Conference on Learning Representations*. [Online]. Available: https://openreview.net/forum?id=rJl-b3RcF7

[17] J. Lin, L. Zhu, W.-M. Chen, W.-C. Wang, and S. Han, "Tiny Machine Learning: Progress and Futures [Feature]," vol. 23, no. 3, pp. 8–34. [Online]. Available: https://ieeexplore.ieee.org/document/10284551

[18] U. Evci, Y. Ioannou, C. Keskin, and Y. Dauphin, "Gradient Flow in Sparse Neural Networks and How Lottery Tickets Win," vol. 36, no. 6, pp. 6577–6586. [Online]. Available: https://ojs.aaai.org/index.php/AAAI/article/view/20611

[19] M. Tan and Q. Le, "EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks," in *Proceedings of the 36th International Conference on Machine Learning*. PMLR, pp. 6105–6114. [Online]. Available: https://proceedings.mlr.press/v97/tan19a.html

[20] C. Wang, G. Zhang, and R. Grosse, "Picking Winning Tickets Before Training by Preserving Gradient Flow," in *International Conference on Learning Representations*. [Online]. Available: https://openreview.net/forum?id=SkgsACVKPH

[21] U. Evci, T. Gale, J. Menick, P. S. Castro, and E. Elsen, "Rigging the lottery: Making all tickets winners," in *Proceedings of the 37th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, H. D. III and A. Singh, Eds., vol. 119. PMLR, pp. 2943–2952. [Online]. Available: https://proceedings.mlr.press/v119/evci20a.html

[22] J. Frankle *et al.*, "Linear Mode Connectivity and the Lottery Ticket Hypothesis," in *Proceedings of the 37th International Conference on Machine Learning*. PMLR, pp. 3259–3269. [Online]. Available: https://proceedings.mlr.press/v119/frankle20a.html

[23] R. Entezari, H. Sedghi, O. Saukh, and B. Neyshabur, "The Role of Permutation Invariance in Linear Mode Connectivity of Neural Networks," in *International Conference on Learning Representations*. [Online]. Available: https://openreview.net/forum?id=dNigytemkL

[24] S. Han, J. Pool, J. Tran, and W. Dally, "Learning both Weights and Connections for Efficient Neural Network," in *Advances in Neural Information Processing Systems*, vol. 28. Curran Associates, Inc. [Online]. Available: https://papers.nips.cc/paper/2015/hash/ae0eb3eed39d2bcef4622b2499a05fe6-Abstract.html

[25] T. Gale, E. Elsen, and S. Hooker. The State of Sparsity in Deep Neural Networks. [Online]. Available: http://arxiv.org/abs/1902.09574

[26] Y. Li, S. Lin, J. Liu, Q. Ye, M. Wang, F. Chao, F. Yang, J. Ma, Q. Tian, and R. Ji, "Towards Compact CNNs via Collaborative Compression," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 6438–6447. [Online]. Available: https://openaccess.thecvf.com/content/CVPR2021/html/Li_Towards_Compact_CNNs_via_Collaborative_Compression_CVPR_2021_paper.html

[27] A. Neelakantan, L. Vilnis, Q. V. Le, I. Sutskever, L. Kaiser, K. Kurach, and J. Martens. Adding Gradient Noise Improves Learning for Very Deep Networks. [Online]. Available: http://arxiv.org/abs/1511.06807

[28] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II," vol. 6, no. 2, pp. 182–197. [Online]. Available: https://ieeexplore.ieee.org/document/996017

[29] M. Wortsman, G. Ilharco, S. Y. Gadre, R. Roelofs, R. Gontijo-Lopes, A. S. Morcos, H. Namkoong, A. Farhadi, Y. Carmon, S. Kornblith, and L. Schmidt, "Model soups: Averaging weights of multiple fine-tuned models improves accuracy without increasing inference time," in *Proceedings of the 39th International Conference on Machine Learning*. PMLR, Jun. 2022, pp. 23 965–23 998.

[30] B. Li, B. Wu, J. Su, and G. Wang, "EagleEye: Fast Sub-net Evaluation for Efficient Neural Network Pruning," in *Computer Vision – ECCV 2020*, A. Vedaldi, H. Bischof, T. Brox, and J.-M. Frahm, Eds. Springer International Publishing, vol. 12347, pp. 639–654. [Online]. Available: https://link.springer.com/10.1007/978-3-030-58536-5_38