Visual Prompting Upgrades Neural Network Sparsification: A Data-Model Perspective

Can Jin[1*], Tianjin Huang[2,3*], Yihua Zhang[4], Mykola Pechenizkiy[2], Sijia Liu[4], Shiwei Liu[5], Tianlong Chen[6]

[1]Rutgers University
[2]Eindhoven University of Technology
[3]University of Exeter,
[4]Michigan State University
[5]University of Oxford
[6]University of North Carolina at Chapel Hill

Abstract

The rapid development of large-scale deep learning models questions the affordability of hardware platforms, which necessitates the pruning to reduce their computational and memory footprints. Sparse neural networks as the product, have demonstrated numerous favorable benefits like low complexity, undamaged generalization, *etc*. Most of the prominent pruning strategies are invented from a *model-centric* perspective, focusing on searching and preserving crucial weights by analyzing network topologies. However, the role of data and its interplay with model-centric pruning has remained relatively unexplored. In this research, we introduce a novel *data-model co-design* perspective: to promote superior weight sparsity by learning important model topology and adequate input data in a synergetic manner. Specifically, customized **V**isual **P**rompts are mounted to upgrade neural **N**etwork **s**parsification in our proposed **VPNs** framework. As a pioneering effort, this paper conducts systematic investigations about the impact of different visual prompts on model pruning and suggests an effective joint optimization approach. Extensive experiments with 3 network architectures and 8 datasets evidence the substantial performance improvements from **VPNs** over existing start-of-the-art pruning algorithms. Furthermore, we find that subnetworks discovered by **VPNs** from pre-trained models enjoy better transferability across diverse downstream scenarios. These insights shed light on new promising possibilities of data-model co-designs for vision model sparsification.

Introduction

Large-scale neural networks like vision and language models (Brown et al. 2020; Touvron et al. 2023) have attracted stupendous attention in nowadays deep learning community, which pose significantly increased demands to computing resources. While remarkable performance has been offered, they suffer from prohibitively high training and inference costs, and deployment of these gigantic models entails substantial memory and computational overhead. For instance, inferring the GPT-3 with 175B parameters requires at least five 80GB A100 GPUs (Frantar and Alistarh 2023).

To establish economic and lightweight network alternatives, model compression serves as an effective tool, gaining great popularity (Dettmers et al. 2022; Frantar and Alistarh 2023). Among plenty of efforts for compression, model pruning (LeCun, Denker, and Solla 1989; Frankle and Carbin 2018) is one of the dominant approaches, aiming to trim down the least significant weights without hurting model performance. It is usually applied subsequent to the convergence of training (Chen et al. 2020), during training process (Chen et al. 2021a), and even prior to the initiation of training (Lee, Ajanthan, and Torr 2019). The resulting sparsity ranges from fine-grained elements like individual weights (Zhu and Gupta 2017) to coarse-grained structures such as neurons (Hu et al. 2016) and blocks (Lagunas et al. 2021). It is worth mentioning that the majority, if not all, of the conventional pruning algorithms, produce sparse neural networks in a *model-centric* fashion – analyzing architecture topologies and capturing their key subset by learning parameterized weight masks (Sehwag et al. 2020) or calculating proxy heuristics based on training dynamics (Han, Mao, and Dally 2015), architecture properties (Hoang et al. 2023), *etc*.

Thanks to the recent advances in large language models (LLMs), the *data-centric* AI regains a spotlight. Techniques like in-context learning (Brown et al. 2020) and prompting (Liu et al. 2023a) construct well-designed prompts or input templates to empower LLMs and reach impressive performances on a variety of tasks. It evidences that such data-centric designs effectively extract and compose knowledge in learned models (Wei et al. 2022), which might be a great assistant to locating critical model topologies. Nevertheless, the influence of data-centric methods on network sparsification has been less studied. To our best knowledge, only one concurrent work (Xu et al. 2023) has explored the possibility of learning *post-pruning prompts* to recover compressed LLMs. Thus, We focus on a different aim:

How to leverage prompts to upgrade vision model sparsification, from a data-model perspective?

Note that the effect of visual prompts on sparse vision models remains mysterious. Also, visual prompts are inherently

*These authors contributed equally.

more complex to comprehend and typically pose greater challenges in terms of both design and learning, in comparison to their linguistic counterparts.

To answer the above research questions, we start with a systematical pilot investigation of existing post-pruning prompts (Xu et al. 2023) on sparse vision models. As presented in Figure 1, **directly inserting *post-pruning* visual prompts into sparse vision models does not necessarily bring performance gains.**

To unlock the capacity of visual prompts in sparse vision models, we propose a *data-model co-design* paradigm. Specifically, we propose **VPNs** (**V**isual **P**rompting Upgrades **N**etworks **S**parsification) that co-trains the visual prompts with parameterized weight masks, exploring superior subnetworks. Our efforts are unfolded with the following five thrusts:

- ⋆ (A Pilot Study) We conduct a pilot study of post-pruning prompts in sparse vision models and surprisingly find its inefficacy in improving the performance of well fine-tuned sparse vision models.
- ⋆ (Algorithm) To unlock the potentials of visual prompts in vision model sparsification, we propose a novel *data-model co-design* sparsification paradigm, termed **VPNs**, which simultaneously optimizes weight masks and tailored visual prompts.
- ⋆ (Experiments) We conduct extensive experiments across diverse datasets, architectures, and pruning regimes. Empirical results consistently highlight the impressive advancement of both performance and efficiency brought by **VPNs**. For example, **VPNs** outperforms the previous state-of-the-art (SoTA) methods {HYDRA (Sehwag et al. 2020), BiP (Zhang et al. 2022), LTH (Chen et al. 2021b)} by $\{3.41\%, 1.69\%, 2.00\%\}$ at 90% sparsity with ResNet-18 on Tiny-ImageNet.
- ⋆ (Extra Findings) More interestingly, we demonstrate that the sparse masks from our **VPNs** enjoy enhanced transferability across multiple downstream tasks.
- ⋆ (Potential Practical Benefits) **VPNs** can be seamlessly integrated into structured pruning approaches, enabling more real-time speedups and memory reduction with competitive accuracies.

Related Works and A Pilot Study

Neural Network Pruning. Pruning (Mozer and Smolensky 1989; LeCun, Denker, and Solla 1989) aims at compressing networks by removing the least important parameters in order to benefit the model generalization (Chen et al. 2022b), robustness (Sehwag et al. 2020), transferability (Chen et al. 2020), *et al*. In the literature, an unpruned network is often termed the "dense network", while its compressed counterpart is referred to as a "subnetwork" of the dense network (Chen et al. 2021b). A commonly adopted compression strategy follows a three-phase pipeline: pre-training, pruning, and fine-tuning. Categorizing based on this pipeline, pruning algorithms can be segmented into post-training pruning, during-training pruning, and prior-training pruning. *Post-training* pruning methods, applied after the dense network converges, are extensively explored. In general, these methods fall under three primary umbrellas: weight magnitude-based techniques (Han, Mao, and Dally 2015), gradient-centric methods (Molchanov et al. 2016), and approaches leveraging Hessians (Hassibi and Stork 1992). Along with the rising of foundational models, more innovative post-training pruning methods have emerged to amplify their resource-efficiency (Zafrir et al. 2021). *During-training* pruning, which is introduced by (Finnoff, Hergert, and Zimmermann 1993), presents an effective variant for model sparsification. It begins by training a dense model and then iteratively trims it based on pre-defined criteria, until obtaining the desired sparse subnetwork. Significant contributions to this approach category are evident in works such as (Chen et al. 2022a). As a more intriguing yet challenging alternative, *prior-training* pruning thrives (Huang et al. 2023; Jaiswal et al. 2023), which targets to identify the optimal subnetwork before fine-tuning the dense network. Mocanu et al. (2018); Dettmers and Zettlemoyer (2019) take a step further to advocate one particular group of sparse neural networks that are extracted at random initialized models, trained from the ground up, and able to reach commendable results.

Prompting. Traditionally, the quest for peak performance is centered on manipulating model weights. However, prompting heralds a pivotal shift towards *data-centric* studies, illuminating the potential of innovative input design. The concept of prompting emerges in the domain of natural language processing (NLP) as a proficient approach for adapting pre-trained models to downstream tasks (Liu et al. 2023a). Specifically, GPT-3 showcases its generalization to downstream transfer learning tasks when equipped with handpicked text prompts, especially in settings like few-shot or zero-shot learning (Brown et al. 2020). There is a significant amount of work around refining text prompting, both in terms of crafting superior prompts (Shin et al. 2020) and representing prompts as task-specific continuous vectors (Liu et al. 2021). The latter involves optimizing these prompts using gradients during the fine-tuning phase, which is termed Prompt Tuning (Li and Liang 2021; Lester, Al-Rfou, and Constant 2021). Interestingly, this approach rivals the performance of full fine-tuning but enjoys the advantage of high parameter efficiency and low storage cost. The design philosophy of prompt tuning is extended to the computer vision realm by Bahng et al. (2022), which incorporates prompt parameters directly into input images, thereby crafting a prompted image, termed Visual Prompt (VP). Building on this foundation, Jia et al. (2022) proposes a visual-prompt tuning method that modifies pre-trained Vision Transformer models by integrating selected parameters into the Transformer's input space. Chen et al. (2023) reveals the importance of correct label mapping between the source and the target classes and introduces iterative label mapping to help boost the performance of VP. Further advancements are made by Liu et al. (2023c); Zheng et al. (2022), which devise a prompt adapter towards enhancing or pinpointing an optimal prompt for a given domain. In a parallel approach, Zang et al. (2022) and Zhou et al. (2022) introduce visual prompts in conjunction with text prompts to

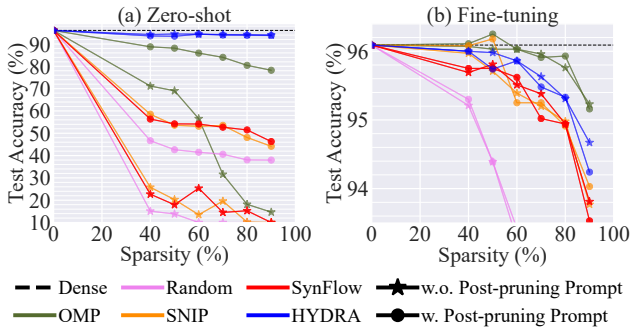

Figure 1: Post-pruning Prompt Results. Performance of 5 pruning methods and their post-pruning prompt counterparts on ResNet-18 and CIFAR10, which are marked as • and ★, respectively. The dashed line indicates the dense network's performance. (a) Post-pruning with zero-shot. (b) Post-pruning with fine-tuning. *Post-pruning prompt is only valid without fine-tuning.*

vision-language models, resulting in a noted improvement in downstream performance.

A Pilot Study

Motivation. The question of whether pruning should be either a more *model-centric* or *data-centric* process continues to be debated within the field. Certain proponents suggest pruning as *model-centric*, with their assertions bolstered by the success of approaches like SynFlow (Tanaka et al. 2020) which, despite not using any real data pass, deliver performances akin to dense networks. Yet, a considerable body of research contradicts this, emphasizing the superiority of post-training pruning techniques over prior-training ones, thereby articulating pruning's dependence on data (Liu et al. 2023b). To further complicate matters, the rise of LLMs has underscored the central role of data in shaping NLP's evolution. New strategies like in-context learning and prompting, designed to enhance LLMs' task-specific performance, have come to the fore. However, the precise role of *data-centric* designs in sparsification remains underexplored, meriting further attention.

To the best of our knowledge, Xu et al. (2023) is the sole concurrent study to delve into the potential of harnessing prompts to recover compressed LLMs. This research illuminates the efficacy of *post-pruning* prompts, both manually crafted and learned "soft" prompts, in enhancing the performance of compressed LLMs. However, the influence of VP on vision model sparsification presents an enigma, as VP is inherently more intricate and poses distinct challenges in designing and learning relative to their textual counterparts. To demystify it, we first investigate the post-pruning prompts on sparsified vision models. The experiments are conducted on ImageNet-1K pre-trained ResNet-18 (He et al. 2016) and CIFAR100 (Krizhevsky, Hinton et al. 2009). We adopt 5 pruning methods, *i.e.*, Random (Liu et al. 2022), OMP (Han, Mao, and Dally 2015), SNIP (Lee, Ajanthan, and Torr 2019), SynFlow (Tanaka et al. 2020), and HYDRA, to analyze the performance of post-pruning prompts across various sparsity levels. To make a holistic study, we apply the post-pruning prompt to the sparse models with and without fine-tuning the subnetwork, referred to as **"Zero-shot"** and **"Fine-tuning"**, respectively. As shown in Figure 1, we find that: Post-pruning prompts only escalate the subnetworks before fine-tuning and bring marginal gains to the subnetwork with fine-tuning. The reason is likely that, after fine-tuning, the sparse model is sufficiently strong, leaving less room for prompts to enhance its performance. Neither of these settings consistently surpasses standard no-prompting approaches, which involve pruning and fine-tuning.

Open Question. As deliberated, the post-pruning prompting paradigm falls short in improving sparse vision models. This situation compels us to ask – *how to effectively utilize visual prompts to enhance the sparsification of vision models?* Our answer: a **data-model co-design** paradigm.

Methodology

In this section, we provide details about **`VPNs`**, which contains (1) designing appropriate visual prompts and (2) incorporating VPs to upgrade the sparse training of vision models in a **data-model** jointly optimization manner. An overview of our proposed **`VPNs`** is depicted in Figure 2.

Designing Appropriate Visual Prompts

Visual prompts are proposed to address the problem of adapting a pre-trained source model to downstream tasks without any task-specific model modification, *e.g.* fine-tuning network weights. To be specific, VP modifies the input image by injecting a small number of learnable parameters. Let $\mathcal{D} = \{(\mathbf{x}_1, y_1), ..., (\mathbf{x}_n, y_n)\}$ denotes the vanilla downstream image dataset, $\mathbf{x}$ is an original image in $\mathcal{D}$ with y as its label, and n represents the total number of images. The generic form of input prompting is then formulated as:

$$\mathbf{x}'(\boldsymbol{\delta}) = h(\mathbf{x}, \boldsymbol{\delta}), \mathbf{x} \in \mathcal{D} = \{(\mathbf{x}_1, y_1), ..., (\mathbf{x}_n, y_n)\}, \quad (1)$$

where $h(\cdot, \cdot)$ is an input transformation that integrates $\mathbf{x}$ with the learnable input perturbation $\boldsymbol{\delta}$ and $\mathbf{x}'$ is the modified data after prompting.

Our VP design first resizes the original image $\mathbf{x}$ to a specific **input size** $i \times i$ and pad it to 224×224 with 0 values to get the resized image. We mark this process as $r^i(\mathbf{x})$, where $r(\cdot)$ refers to the resize and pad operation and i indicates the target size, *i.e.* input size. Subsequently, we initiate the perturbation parameters of $\boldsymbol{\delta}$ as a 224×224 matrix and mask a portion of them. Different visual prompts can be crafted by masking parameters in diverse shapes, locations, and sizes. In our case, the fixed mask is a central square matrix and the left four peripheral segments stay tunable. This kind of perturbation design is similar to *pad prompt* in Bahng et al. (2022) and the width of each peripheral side marked as p is called **pad size**. More details about the prompt can be found in Appendix. Finally, the input prompting operation of **`VPNs`** is described as:

$$\mathbf{x}'(\boldsymbol{\delta}) = h(\mathbf{x}, \boldsymbol{\delta}) = r^i(\mathbf{x}) + \boldsymbol{\delta}^p, \mathbf{x} \in \mathcal{D}, \quad (2)$$

where $\boldsymbol{\delta}^p$ is the pad prompt perturbation with a pad size of p. Note that, usually, $i + 2p$ is larger than the input sample size like 224 to sufficiently utilize all sample pixels.

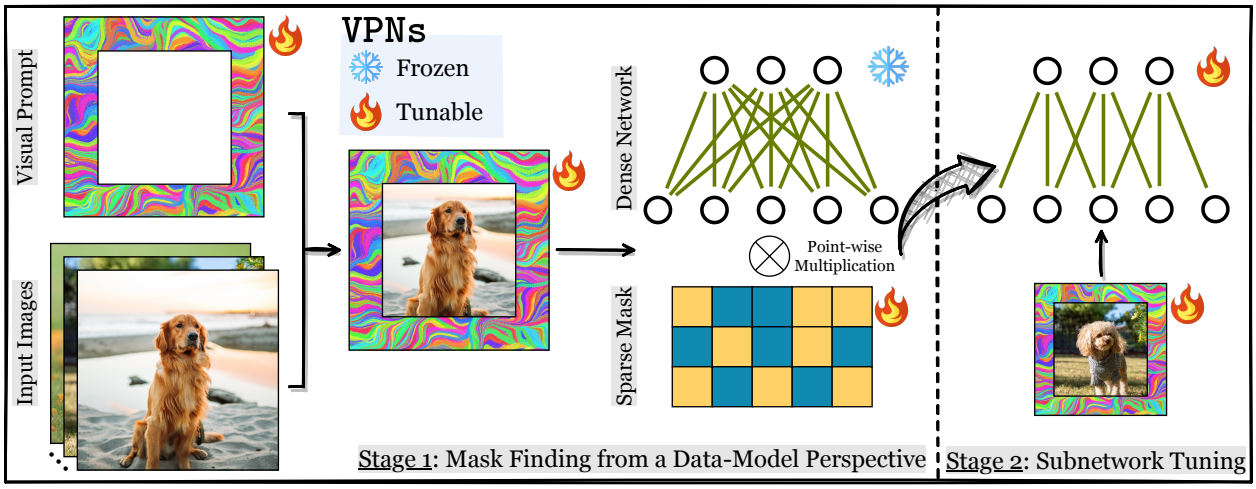

Figure 2: Overview of **VPNs**. In stage 1, it locates sparse topologies from a data-model perspective. A tailored VP is added to the input and weight masks are jointly optimized with VP. In stage 2, the identified subnetwork is further fine-tuned with its VP.

Upgrading Network Sparsification with VP

Given the input prompt formulation (Equation 2), VP seeks to advance downstream task performance of a pre-trained source model $f_{\boldsymbol{\theta}_{\text{pre}}}$ by optimizing the tunable part in $\boldsymbol{\delta}$. Here $\boldsymbol{\theta}_{\text{pre}}$ refers to the pre-trained weights that are fixed in this stage. It raises a *prompt optimization problem* as follows:

$$\underset{\boldsymbol{\delta}}{\text{minimize}} \quad \mathbb{E}_{(\mathbf{x},y)\in\mathcal{D}}\mathcal{L}(f_{\boldsymbol{\theta}_{\text{pre}}}(\mathbf{x}'(\boldsymbol{\delta})), y), \tag{3}$$

where $\mathcal{L}$ is the objective function such as a cross-entropy loss for image recognition problems. As for the network sparsification, we recast it as an empirical risk minimization with respect to a learnable parameterized mask and the corresponding model weights can be frozen. Then a *mask finding problem* is depicted as below:

$$\begin{aligned} \underset{\mathbf{m}}{\text{minimize}}\ & \mathbb{E}_{(\mathbf{x},y)\in\mathcal{D}}\mathcal{L}(f_{\boldsymbol{\theta}_{\text{pre}}\odot\mathbf{m}}(\mathbf{x}), y), \\ & s.t.\ ||\mathbf{m}||_0 \leq (1-s)|\boldsymbol{\theta}_{\text{pre}}|, \end{aligned} \tag{4}$$

where $\mathbf{m}$ is the mask variable, $\boldsymbol{\theta}_{\text{pre}} \odot \mathbf{m}$ is a point-wise multiplication between the mask and model weights, s denotes the desired sparsity level, and $|\boldsymbol{\theta}_{\text{pre}}|$ refers to the number of parameters in $\boldsymbol{\theta}_{\text{pre}}$.

Our proposed **VPNs** leverages visual prompts to upgrade the process of model sparsification by seamlessly integrating Equations 3 and 4. To be specific, the joint optimization of prompt $\boldsymbol{\delta}$ and $\mathbf{m}$ is described as follows:

$$\begin{aligned} \underset{\mathbf{m},\boldsymbol{\delta}}{\text{minimize}}\ & \mathbb{E}_{(\mathbf{x},y)\in\mathcal{D}}\mathcal{L}(f_{\boldsymbol{\theta}_{\text{pre}}\odot\mathbf{m}}(\mathbf{x}'(\boldsymbol{\delta})), y) \\ & s.t.\ ||\mathbf{m}||_0 \leq (1-s)|\boldsymbol{\theta}_{\text{pre}}|, \end{aligned} \tag{5}$$

where the mask $\mathbf{m}$ will be turned into a binary matrix. The thresholding technique from Ramanujan et al. (2020) is applied to map large and small scores to 1 and 0, respectively.

After obtaining sparse subnetworks from **VPNs**, a subsequent retraining phase is attached. It is another data-model co-optimization problem of VP and model weights as below:

$$\underset{\boldsymbol{\delta},\boldsymbol{\theta}}{\text{minimize}}\ \mathbb{E}_{(\mathbf{x},y)\in\mathcal{D}}\mathcal{L}(f_{\boldsymbol{\theta}\odot\mathbf{m}}(\mathbf{x}'(\boldsymbol{\delta})), y) \quad s.t.\ \mathbf{m} = \mathbf{m}_s, \tag{6}$$

where $\boldsymbol{\theta}$ is model parameters that are initialized as $\boldsymbol{\theta}_{\text{pre}}$. $\mathbf{m}_s$ is the mask found by Equation 5, and is fixed in this stage.

Overall Procedure of VPNs

Our **VPNs** first creates a VP following Equation 2. Then, to locate the **VPNs** sparse subnetwork, VP and the parameterized mask are jointly optimized based on Equation 5. $\mathbf{m}$ is initialized with a scaled-initialization from Sehwag et al. (2020), $\boldsymbol{\delta}$ adopts a 0 initialization, and $\boldsymbol{\theta}$ is initialized with $\boldsymbol{\theta}_{\text{pre}}$ which stays frozen. Finally, the weights of found sparse subnetwork are further fine-tuned together with VP, as indicated in Equation 6. During this step, $\boldsymbol{\theta}$ is initialized with $\boldsymbol{\theta}_{\text{pre}}$, visual prompt $\boldsymbol{\delta}$ and mask $\mathbf{m}$ inherit the value of $\boldsymbol{\delta}_s$ and $\mathbf{m}_s$ from the previous stage, respectively. Note that here $\mathbf{m}$ is kept frozen. The detailed procedure of **VPNs** is summarized in the Algorithm in the Appendix. It is worth mentioning that such *data-model* co-design, *i.e.*, **VPNs**, presents a greatly improved efficiency in terms of searching desired high-quality subnetworks. For instance, compared to previous *model-centric* approaches, **VPNs** only needs half the epochs of HYDRA and OMP, while achieving better accuracy (see Appendix).

Experiments

To evaluate the effectiveness of our prompting-driven sparsification method, we follow the most common evaluation of visual prompting, i.e., evaluating sparse models pre-trained on a large dataset (ImageNet-1K) on various visual domains. Moreover, we conduct extensive empirical experiments including (1) Affirming the superior performance of **VPNs** over different datasets and architectures; (2) The transferability of **VPNs** across different datasets is investigated; (3) We further analyze the computational complexity of **VPNs** through the lens of time consumption, training epochs, and gradient calculating steps; (4) Our study also encompasses in-depth investigations into structured pruning algorithms; (5) Ablation studies are presented, which concentrate on the

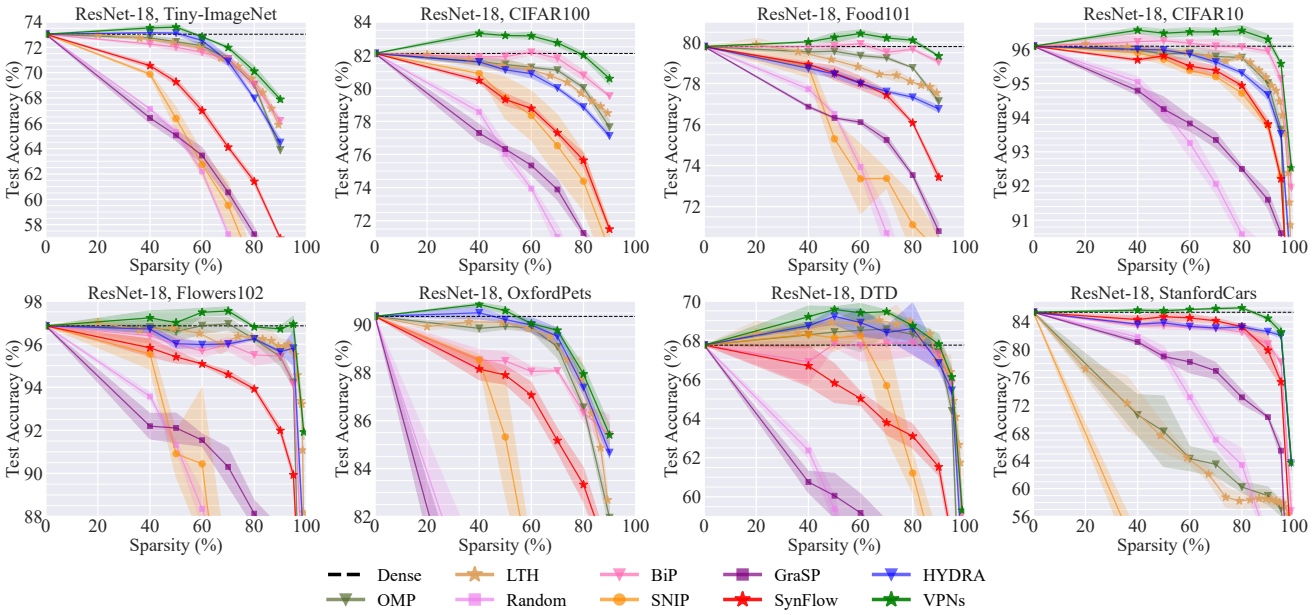

Figure 3: Downstream Fine-tuning Results. The performance overview of 9 unstructured pruning algorithms. All the models are pre-trained on ImageNet-1K; and then pruned and fine-tuned both on the specific downstream dataset. The performance of the dense model and VPNs' best are marked using dashed lines. All the results are averaged over 3 runs.

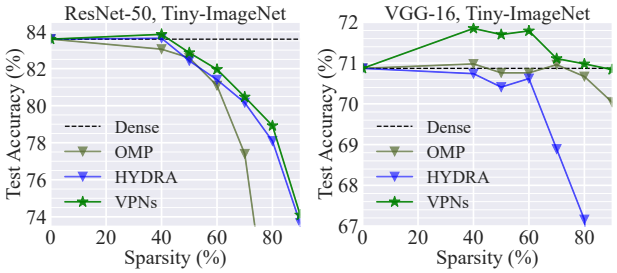

Figure 4: Downstream Fine-tuning Results. The performance overview of VPNs, HYDRA, and OMP. All the results are obtained with ImageNet-1K pre-trained ResNet-50 and VGG-16, fine-tuned on Tiny-ImageNet.

influence of different VP methods, pad and input sizes.

Implementation Details

Network and Datasets. We use three pre-trained network architectures for our experiments – ResNet-18 (He et al. 2016), ResNet-50 (He et al. 2016), and VGG-16 (Simonyan and Zisserman 2014), which can be downloaded from official Pytorch Model Zoo[1]. These models are pre-trained on the ImageNet-1K (Deng et al. 2009). We then evaluate the effectiveness of VPNs over **eight** downstream datasets – Tiny ImageNet (Le and Yang 2015), StanfordCars (Krause et al. 2013), OxfordPets (Parkhi et al. 2012), Food101 (Bossard, Guillaumin, and Van Gool 2014), DTD (Cimpoi et al. 2014), Flowers102 (Nilsback and Zisserman 2008), CIFAR10/100 (Krizhevsky, Hinton et al. 2009), respectively. Further details of datasets are in the Appendix.

Pruning Baselines. We select **eight** representative state-of-the-art (SoTA) pruning algorithms as our baselines. (1) *Random Pruning* (Random) (Liu et al. 2022) is commonly used as a basic sanity check in pruning studies. (2) *One-shot Magnitude Pruning* (OMP) removes weights with the globally smallest magnitudes (Han, Mao, and Dally 2015). (3) *The lottery ticket hypothesis* (LTH) iteratively prunes the 20% of remaining weights with the globally least magnitudes and rewinds model weights to their initial state. In our experiments, weights are rewound to their ImageNet-1K pre-trained weights, following the default configurations in Chen et al. (2021b). (4) *Pruning at initialization* (PaI) locates sparse subnetworks at the initialization phase by the defined salience metric. We opt for three widely-recognized methodologies: SNIP (Lee, Ajanthan, and Torr 2019), GraSP (Wang et al. 2020), and SynFlow (Tanaka et al. 2020) (5) *HYDRA* (Sehwag et al. 2020) prunes weights based on the least importance scores, which is the most important baseline as it can be seen as our method without the visual prompt design. (6) *BiP* (Zhang et al. 2022), characterized as a SoTA pruning algorithm, formalizes the pruning process within a bi-level optimization framework.

Training and Evaluation. We follow the pruning baselines implementation in (Liu et al. 2022), selecting optimal hyper-parameters for various pruning algorithms by grid search. As for visual prompts, our default VP design in **VPNs** employs a pad prompt with an input size of 224 and a pad size of 16. We also use an input size of 224 for all the

[1]https://pytorch.org/vision/stable/models.html

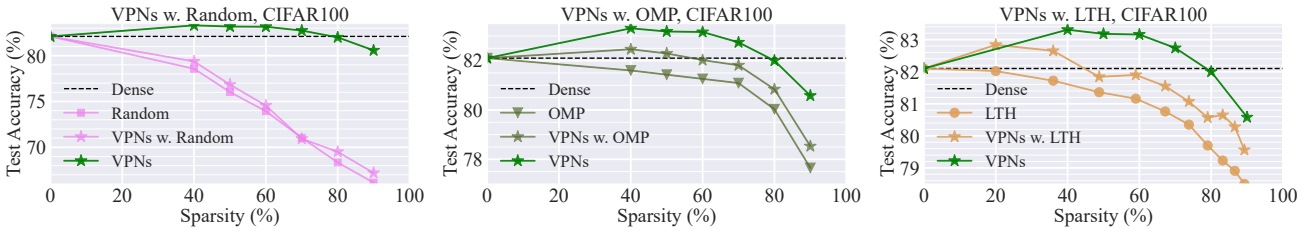

Figure 5: VPNs Paradigm Applied to Current Methods. The performance overview of VPNs pruning paradigm applied to Random, OMP, and LTH pruning named VPNs w. Random, VPNs w. OMP, and VPNs w. LTH. Results are based on ResNet-18 pre-trained on ImageNet-1K and fine-tuned on CIFAR100. VPNs paradigm advances Random, OMP, and LTH consistently.

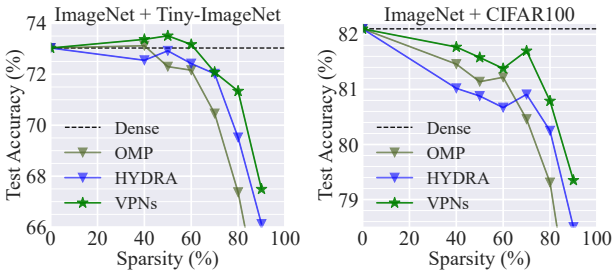

Figure 6: ImagetNet Mask Finding and Downstream Subnetwork Tuning Results. The performance overview of VPNs, HYDRA, and OMP. The models are pruned on ImageNet-1K and fine-tuned on Tiny-ImageNet and CIFAR100.

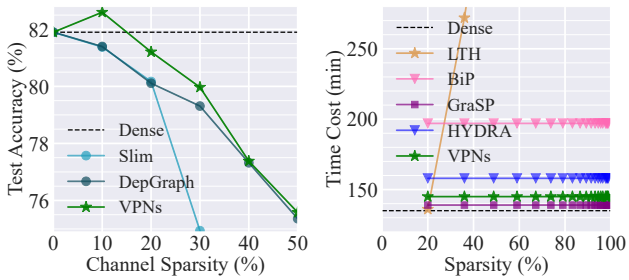

Figure 7: Structured Pruning Results (left) and Time Consumption (right). Results are based on ImageNet pre-trained ResNet-18, fine-tuned on CIFAR-100.

baselines to ensure a fair comparison. More implementation details are in the Appendix.

Main Results

Superior Performance of VPNs. Using a ResNet-18 pre-trained on ImageNet-1K, we evaluate the capability of VPNs in pruning models across multiple downstream datasets. As illustrated in Figure 3, several positive observations can be drawn: ❶ The dominance of VPNs is especially pronounced on larger datasets such as Tiny-ImageNet, CIFAR100, Food101, and CIFAR10. At 90% sparsity level, VPNs outperforms {HYDRA, BiP, LTH} by {3.41%, 1.69%, 2.00%} on Tiny-ImageNet and surpasses {HYDRA, BiP, OMP} by {3.46%, 2.06%, 2.93%} on CIFAR100. ❷ VPNs still delivers top-tier results on smaller datasets like Flowers102, OxfordPets, DTD, and StanfordCars. For instance, the test accuracy of VPNs is {1.12%, 2.79%, 2.71%} higher than {HYDRA, BiP, OMP} at 95% sparsity on Flowers102. ❸ VPNs outperforms fully fine-tuned dense models at high sparsity levels on all eight downstream datasets. It finds subnetworks better than dense counterparts at {50%, 70%, 80%, 90%} sparsity on {Tiny-ImageNet, CIFAR100, Food101, CIFAR10} and {70%, 50%, 90%, 90%} sparsity on {Flowers102, OxfordPets, DTD, StanfordCars}.

We conduct additional experiments with ResNet-50 and VGG-16 to investigate the performance of VPNs over different architectures. These models are pre-trained on ImageNet-1K and fine-tuned on Tiny-ImageNet. All pruning methods are applied in the fine-tuning stage. As shown in Figure 4, VPNs reaches outstanding performance across diverse architectures consistently, compared to OMP ($0.85\% \sim 12.23\%$ higher accuracy on ResNet-50) and HYDRA ($1.14\% \sim 4.08\%$ higher accuracy on VGG-16). It's noteworthy to highlight that OMP and HYDRA represent the most prominent baselines according to from Figure 3.

Additional Investigation and Ablation Study

Transferability of VPNs. Meanwhile, we investigate the transferability of subnetworks identified by VPNs across diverse downstream tasks. We apply VPNs, HYDRA, and OMP pruning on ResNet-18 and ImageNet-1K to identify subnetworks, subsequently fine-tune them on CIFAR100 and Tiny-ImageNet separately. The results are depicted in Figure 6, it can be observed that: ❶ VPNs consistently excels over SoTA algorithms across multiple datasets. At an 80% sparsity level on Tiny-ImageNet, VPNs has {3.97%, 1.57%} higher test accuracy than {OMP, HYDRA}. Moreover, VPNs outperforms {OMP, HYDRA} by {2.75%, 0.80%} at 90% sparsity on CIFAR100. ❷ VPNs subnetworks can surpass the dense network on specific datasets. At 60% sparsity on Tiny-ImageNet, subnetworks identified by VPNs have better performance than their dense counterparts. Consequently, VPNs has transferability over datasets.

Superiority of VPNs Pruning Paradigm. Furthermore, we endeavor to explore the potential of the VPNs pruning paradigm to enhance the effect of existing pruning algorithms. We integrate the VPNs pruning paradigm with Random, OMP, and LTH pruning, forming VPNs w. Random,

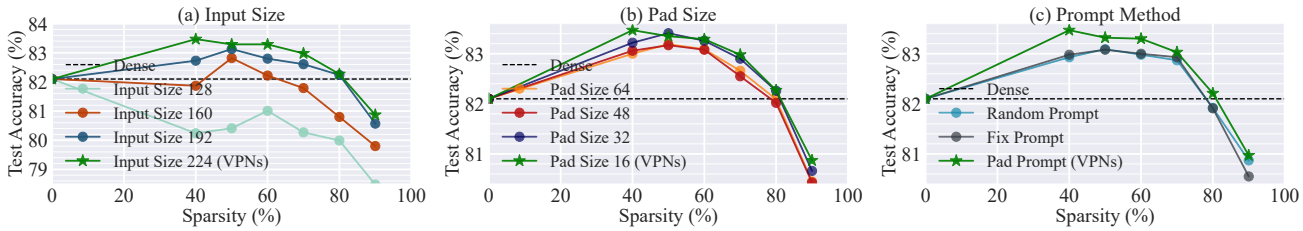

Figure 8: Ablation of VP Designs. Results based on ImageNet pre-trained ResNet-18 and fine-tuned on CIFAR100. (a) Vary input size with pad size of 16. (b) Vary pad size with input size of 224. (c) Vary VP method with 13K prompt parameters.

VPNs w. OMP, and VPNs w. LTH respectively. For the purpose of consistency, the VP utilized in the experiment is kept identical to the one used in VPNs. The results are based on ResNet-18 pre-trained on ImageNet-1K and fine-tuned on CIFAR100. As depicted in Figure 5. We observe that VPNs combined with existing prunings consistently surpasses their original counterpart. For example, At 80% sparsity, {VPNs w. Random, VPNs w. OMP, VPNs w. LTH} surpass their corresponding original pruning by {1.16%, 0.81%, 0.79%}.

VPNs for Structured Pruning. To assess the potential of VPNs in structured pruning, we perform an empirical comparison between VPNs and renowned structured pruning techniques such as Slim (Liu et al. 2017) and DepGraph (Fang et al. 2023). The evaluations are conducted using a pre-trained ResNet-18 model on ImageNet-1K, fine-tuned on CIFAR-100. See Appendix for more details. From the results presented in Figure 7 (left), we observe that: ❶ VPNs enjoys superior performance consistently across various channel sparsity levels in comparison to Slim and DepGraph, achieving higher accuracy by $1.04\% \sim 9.54\%$ and $0.02\% \sim 1.20\%$ respectively. ❷ VPNs simultaneously reduces both training and inference FLOPs and memory costs. For example, at 10% and 20% channel sparsity levels, VPNs achieves speedup ratios of $1.1\times$ and $1.3\times$ while reducing memory costs by 15.26% and 31.71% respectively, without compromising the performance relative to the dense network. The speedup ratio is quantified as $\frac{\text{FLOPs(dense)}}{\text{FLOPs(subnetwork)}}$.

Computational Complexity. An effective pruning algorithm should exhibit computational efficiency. Accordingly, we evaluate the computational complexity of VPNs in comparison to SoTA pruning methods. Our criterion contains training time consumption, training epochs, and gradient calculating steps with evaluations conducted on ImageNet-1K pre-trained ResNet-18 and fine-tuned on CIFAR100. Results are displayed in Figure 7 (right) and Table of training epochs in the Appendix, several positive findings can be drawn: ❶ VPNs consistently outperforms BiP and HYDRA in terms of time efficiency, achieving a time reduction of 26% and 8.97% respectively across varying sparsity levels while exhibits a time consumption comparable to GraSP. It is also noteworthy to mention that LTH's time consumption exhibits an exponential increase in relation to sparsity growth. ❷ VPNs requires fewest epochs and steps to attain optimal performance. Specifically, for achieving a 90% sparsity, VPNs requires 95%, 50%, and 50% fewer epochs in comparison to LTH, GraSP, and HYDRA, respectively. Moreover, it demands 90% and 33% fewer steps than LTH and BiP separately.

Ablation – VP Designs. We systematically examine the impact of different VP designs on VPNs. Our experiments are based on pre-trained ResNet-18 and fine-tuned on CIFAR100.

Input Size. We employ pad prompts with a fixed pad size of 16 while varying the input size from 128 to 224 to assess the effect of input size on the performance of VPNs. As illustrated in Figure 8 (a), As the input size increases, we observe a corresponding rise in test accuracy. This underscores the imperative of harnessing the entirety of information available in the original images.

Pad Size. Similarly, to investigate the impact of the pad size, we fix the input size of 224 and vary the pad size of the VP from 16 to 64. The results are shown in 8 (b). Pad sizes 16 and 32 exhibit the best performance and the test accuracy declines as the pad sizes increase further, which indicates that a small number of prompt parameters benefits more to VPNs pruning performance.

Visual Prompt Strategies. We conduct an investigation into three distinct types of VP methods: the pad prompt, the random prompt, and the fix prompt. Figure 8 (c) provides a visual representation of the pad prompt. In contrast, the random prompt is tunable within a randomly chosen square section of the perturbation $\boldsymbol{\delta}$ as defined in Equation 2. The fix prompt, on the other hand, restricts tunability to the top-left square segment of $\boldsymbol{\delta}$. See Appendix for more details. In our experiments, all VP methods are kept consistent with 13K tunable parameters. The results are shown in Figure 8 (c). We observe that the pad prompt outperforms both the fix and random prompts for VPNs.

Conclusion

In this work, we highlight the limitations of post-pruning prompts in enhancing vision subnetworks. To harness the potential of visual prompts for neural network sparsification, we introduce an innovative data-model co-design algorithm, termed **VPNs**. Comprehensive experiments across diverse datasets, architectures, and pruning methods consistently validate the superior performance and efficiency offered by **VPNs**. We further demonstrate the transferability of subnetworks identified by **VPNs** across multiple datasets, emphasizing its practical utility in a broader applications.

References

Bahng, H.; Jahanian, A.; Sankaranarayanan, S.; and Isola, P. 2022. Exploring visual prompts for adapting large-scale models. *arXiv preprint arXiv:2203.17274*, 1(3): 4.

Bossard, L.; Guillaumin, M.; and Van Gool, L. 2014. Food-101–mining discriminative components with random forests. In *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part VI 13*, 446–461. Springer.

Brown, T.; Mann, B.; Ryder, N.; Subbiah, M.; Kaplan, J. D.; Dhariwal, P.; Neelakantan, A.; Shyam, P.; Sastry, G.; Askell, A.; et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33: 1877–1901.

Chen, A.; Yao, Y.; Chen, P.-Y.; Zhang, Y.; and Liu, S. 2023. Understanding and improving visual prompting: A label-mapping perspective. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 19133–19143.

Chen, T.; Cheng, Y.; Gan, Z.; Yuan, L.; Zhang, L.; and Wang, Z. 2021a. Chasing Sparsity in Vision Transformers: An End-to-End Exploration. In Beygelzimer, A.; Dauphin, Y.; Liang, P.; and Vaughan, J. W., eds., *Advances in Neural Information Processing Systems*.

Chen, T.; Frankle, J.; Chang, S.; Liu, S.; Zhang, Y.; Carbin, M.; and Wang, Z. 2021b. The lottery tickets hypothesis for supervised and self-supervised pre-training in computer vision models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 16306–16316.

Chen, T.; Frankle, J.; Chang, S.; Liu, S.; Zhang, Y.; Wang, Z.; and Carbin, M. 2020. The lottery ticket hypothesis for pre-trained bert networks. *Advances in neural information processing systems*, 33: 15834–15846.

Chen, T.; Zhang, Z.; Wang, P.; Balachandra, S.; Ma, H.; Wang, Z.; and Wang, Z. 2022a. Sparsity winning twice: Better robust generalization from more efficient training. *arXiv preprint arXiv:2202.09844*.

Chen, T.; Zhang, Z.; Wu, J.; Huang, R.; Liu, S.; Chang, S.; and Wang, Z. 2022b. Can You Win Everything with A Lottery Ticket? *Transactions on Machine Learning Research*.

Cimpoi, M.; Maji, S.; Kokkinos, I.; Mohamed, S.; and Vedaldi, A. 2014. Describing textures in the wild. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 3606–3613.

Deng, J.; Dong, W.; Socher, R.; Li, L.-J.; Li, K.; and Fei-Fei, L. 2009. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, 248–255. Ieee.

Dettmers, T.; Lewis, M.; Belkada, Y.; and Zettlemoyer, L. 2022. Llm. int8 (): 8-bit matrix multiplication for transformers at scale. *arXiv preprint arXiv:2208.07339*.

Dettmers, T.; and Zettlemoyer, L. 2019. Sparse networks from scratch: Faster training without losing performance. *arXiv preprint arXiv:1907.04840*.

Fang, G.; Ma, X.; Song, M.; Mi, M. B.; and Wang, X. 2023. Depgraph: Towards any structural pruning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 16091–16101.

Finnoff, W.; Hergert, F.; and Zimmermann, H. G. 1993. Improving model selection by nonconvergent methods. *Neural Networks*, 6(6): 771–783.

Frankle, J.; and Carbin, M. 2018. The lottery ticket hypothesis: Finding sparse, trainable neural networks. *arXiv preprint arXiv:1803.03635*.

Frantar, E.; and Alistarh, D. 2023. Massive language models can be accurately pruned in one-shot. *arXiv preprint arXiv:2301.00774*.

Han, S.; Mao, H.; and Dally, W. J. 2015. Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding. *arXiv preprint arXiv:1510.00149*.

Hassibi, B.; and Stork, D. 1992. Second order derivatives for network pruning: Optimal brain surgeon. *Advances in neural information processing systems*, 5.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 770–778.

Hoang, D. N.; Liu, S.; Marculescu, R.; and Wang, Z. 2023. REVISITING PRUNING AT INITIALIZATION THROUGH THE LENS OF RAMANUJAN GRAPH. In *The Eleventh International Conference on Learning Representations*.

Hu, H.; Peng, R.; Tai, Y.-W.; and Tang, C.-K. 2016. Network trimming: A data-driven neuron pruning approach towards efficient deep architectures. *arXiv preprint arXiv:1607.03250*.

Huang, T.; Liu, S.; Chen, T.; Fang, M.; Shen, L.; Menkovski, V.; Yin, L.; Pei, Y.; and Pechenizkiy, M. 2023. Enhancing Adversarial Training via Reweighting Optimization Trajectory. *arXiv preprint arXiv:2306.14275*.

Jaiswal, A.; Liu, S.; Chen, T.; and Wang, Z. 2023. The Emergence of Essential Sparsity in Large Pre-trained Models: The Weights that Matter. *arXiv preprint arXiv:2306.03805*.

Jia, M.; Tang, L.; Chen, B.-C.; Cardie, C.; Belongie, S.; Hariharan, B.; and Lim, S.-N. 2022. Visual prompt tuning. In *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XXXIII*, 709–727. Springer.

Krause, J.; Stark, M.; Deng, J.; and Fei-Fei, L. 2013. 3d object representations for fine-grained categorization. In *Proceedings of the IEEE international conference on computer vision workshops*, 554–561.

Krizhevsky, A.; Hinton, G.; et al. 2009. Learning multiple layers of features from tiny images.

Lagunas, F.; Charlaix, E.; Sanh, V.; and Rush, A. M. 2021. Block pruning for faster transformers. *arXiv preprint arXiv:2109.04838*.

Le, Y.; and Yang, X. 2015. Tiny imagenet visual recognition challenge. *CS 231N*, 7(7): 3.

LeCun, Y.; Denker, J.; and Solla, S. 1989. Optimal brain damage. *Advances in neural information processing systems*, 2.

Lee, N.; Ajanthan, T.; and Torr, P. 2019. SNIP: SINGLE-SHOT NETWORK PRUNING BASED ON CONNECTION SENSITIVITY. In *International Conference on Learning Representations*.

Lester, B.; Al-Rfou, R.; and Constant, N. 2021. The power of scale for parameter-efficient prompt tuning. *arXiv preprint arXiv:2104.08691*.

Li, X. L.; and Liang, P. 2021. Prefix-tuning: Optimizing continuous prompts for generation. *arXiv preprint arXiv:2101.00190*.

Liu, P.; Yuan, W.; Fu, J.; Jiang, Z.; Hayashi, H.; and Neubig, G. 2023a. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. *ACM Computing Surveys*, 55(9): 1–35.

Liu, S.; Chen, T.; Chen, X.; Shen, L.; Mocanu, D. C.; Wang, Z.; and Pechenizkiy, M. 2022. The unreasonable effectiveness of random pruning: Return of the most naive baseline for sparse training. *arXiv preprint arXiv:2202.02643*.

Liu, S.; Chen, T.; Zhang, Z.; Chen, X.; Huang, T.; Jaiswal, A.; and Wang, Z. 2023b. Sparsity May Cry: Let Us Fail (Current) Sparse Neural Networks Together! *arXiv preprint arXiv:2303.02141*.

Liu, W.; Shen, X.; Pun, C.-M.; and Cun, X. 2023c. Explicit visual prompting for low-level structure segmentations. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 19434–19445.

Liu, X.; Ji, K.; Fu, Y.; Tam, W. L.; Du, Z.; Yang, Z.; and Tang, J. 2021. P-tuning v2: Prompt tuning can be comparable to fine-tuning universally across scales and tasks. *arXiv preprint arXiv:2110.07602*.

Liu, Z.; Li, J.; Shen, Z.; Huang, G.; Yan, S.; and Zhang, C. 2017. Learning efficient convolutional networks through network slimming. In *Proceedings of the IEEE international conference on computer vision*, 2736–2744.

Mocanu, D. C.; Mocanu, E.; Stone, P.; Nguyen, P. H.; Gibescu, M.; and Liotta, A. 2018. Scalable training of artificial neural networks with adaptive sparse connectivity inspired by network science. *Nature communications*, 9(1): 2383.

Molchanov, P.; Tyree, S.; Karras, T.; Aila, T.; and Kautz, J. 2016. Pruning convolutional neural networks for resource efficient inference. *arXiv preprint arXiv:1611.06440*.

Mozer, M. C.; and Smolensky, P. 1989. Using relevance to reduce network size automatically. *Connection Science*, 1(1): 3–16.

Nilsback, M.-E.; and Zisserman, A. 2008. Automated flower classification over a large number of classes. In *2008 Sixth Indian conference on computer vision, graphics & image processing*, 722–729. IEEE.

Parkhi, O. M.; Vedaldi, A.; Zisserman, A.; and Jawahar, C. 2012. Cats and dogs. In *2012 IEEE conference on computer vision and pattern recognition*, 3498–3505. IEEE.

Ramanujan, V.; Wortsman, M.; Kembhavi, A.; Farhadi, A.; and Rastegari, M. 2020. What's hidden in a randomly weighted neural network? In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 11893–11902.

Sehwag, V.; Wang, S.; Mittal, P.; and Jana, S. 2020. Hydra: Pruning adversarially robust neural networks. *Advances in Neural Information Processing Systems*, 33: 19655–19666.

Shin, T.; Razeghi, Y.; Logan IV, R. L.; Wallace, E.; and Singh, S. 2020. Autoprompt: Eliciting knowledge from language models with automatically generated prompts. *arXiv preprint arXiv:2010.15980*.

Simonyan, K.; and Zisserman, A. 2014. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*.

Tanaka, H.; Kunin, D.; Yamins, D. L.; and Ganguli, S. 2020. Pruning neural networks without any data by iteratively conserving synaptic flow. *Advances in neural information processing systems*, 33: 6377–6389.

Touvron, H.; Lavril, T.; Izacard, G.; Martinet, X.; Lachaux, M.-A.; Lacroix, T.; Rozière, B.; Goyal, N.; Hambro, E.; Azhar, F.; et al. 2023. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*.

Wang, Z.; Tsvetkov, Y.; Firat, O.; and Cao, Y. 2020. Gradient vaccine: Investigating and improving multi-task optimization in massively multilingual models. *arXiv preprint arXiv:2010.05874*.

Wei, J.; Tay, Y.; Bommasani, R.; Raffel, C.; Zoph, B.; Borgeaud, S.; Yogatama, D.; Bosma, M.; Zhou, D.; Metzler, D.; et al. 2022. Emergent abilities of large language models. *arXiv preprint arXiv:2206.07682*.

Xu, Z.; Liu, Z.; Chen, B.; Tang, Y.; Wang, J.; Zhou, K.; Hu, X.; and Shrivastava, A. 2023. Compress, Then Prompt: Improving Accuracy-Efficiency Trade-off of LLM Inference with Transferable Prompt. *arXiv preprint arXiv:2305.11186*.

Zafrir, O.; Larey, A.; Boudoukh, G.; Shen, H.; and Wasserblat, M. 2021. Prune once for all: Sparse pre-trained language models. *arXiv preprint arXiv:2111.05754*.

Zang, Y.; Li, W.; Zhou, K.; Huang, C.; and Loy, C. C. 2022. Unified vision and language prompt learning. *arXiv preprint arXiv:2210.07225*.

Zhang, Y.; Yao, Y.; Ram, P.; Zhao, P.; Chen, T.; Hong, M.; Wang, Y.; and Liu, S. 2022. Advancing model pruning via bi-level optimization. *Advances in Neural Information Processing Systems*, 35: 18309–18326.

Zheng, Z.; Yue, X.; Wang, K.; and You, Y. 2022. Prompt vision transformer for domain generalization. *arXiv preprint arXiv:2208.08914*.

Zhou, K.; Yang, J.; Loy, C. C.; and Liu, Z. 2022. Conditional prompt learning for vision-language models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 16816–16825.

Zhu, M.; and Gupta, S. 2017. To prune, or not to prune: exploring the efficacy of pruning for model compression. *arXiv preprint arXiv:1710.01878*.