(PASS) Visual Prompt Locates Good Structure Sparsity through a Recurrent HyperNetwork

Tianjin Huang[1,2]*, Yong Tao[1]*, Meng Fang[3], Li Shen[4], Fan Liu[5], Yulong Pei[2]
Mykola Pechenizkiy[2], Tianlong Chen[6]
[1]University of Exeter, [2]Eindhoven University of Technology, [3]University of liverpool
[4]Sun Yat-sen University, [5]Hohai University, [6]UNC-Chapel Hill
{t.huang2, yt438}@exeter.ac.uk, Meng.Fang@liverpool.ac.uk,
mathshenli@gmail.com, fanliu@hhu.edu.cn, {y.pei.1,
M.pechenizkiy}@tue.nl, tianlong@mit.edu

Large-scale neural networks have demonstrated remarkable performance in different domains like vision and language processing, although at the cost of massive computation resources. As illustrated by compression literature, structural model pruning is a prominent algorithm to encourage model efficiency, thanks to its acceleration-friendly sparsity patterns. One of the key questions of structural pruning is how to estimate the channel significance. In parallel, work on data-centric AI has shown that prompting-based techniques enable impressive generalization of large language models across diverse downstream tasks. In this paper, we investigate a charming possibility - *leveraging visual prompts to capture the channel importance and derive high-quality structural sparsity*. To this end, we propose a novel algorithmic framework, namely PASS. It is a tailored hyper-network to take both visual prompts and network weight statistics as input, and output layer-wise channel sparsity in a recurrent manner. Such designs consider the intrinsic channel dependency between layers. Comprehensive experiments across multiple network architectures and six datasets demonstrate the superiority of PASS in locating good structural sparsity. For example, at the same FLOPs level, PASS subnetworks achieve $1\% \sim 3\%$ better accuracy on Food101 dataset; or with a similar performance of 80% accuracy, PASS subnetworks obtain $0.35\times$ more speedup than the baselines.

1. Introduction

Recently, large-scale neural networks, particularly in the field of vision and language modeling, have received upsurging interest due to the promising performance for both natural language [1, 2] and vision tasks [3, 4]. While these models have delivered remarkable performance, their colossal model size, coupled with their vast memory and computational requirements, pose significant obstacles to model deployment. To solve this daunting challenge, model compression techniques have re-gained numerous attention [5, 6] and knowledge distillation can be further adopted on top of them to recover optimal performance [7, 8]. Among them, model pruning is a well-established method known for its capacity to reduce model size without compromising performance and structural model pruning has garnered significant interest due to its ability to systematically eliminate superfluous structural components, such as entire neurons, channels, or filters, rather than individual weights, making it more hardware-friendly [9, 10].

In the context of structural pruning for *vision models*, the paramount task is the estimation of the importance of each structure component, such as channel or filters. It is a fundamental challenge since it requires dissecting the neural network behavior and a precise evaluation of the relevance of individual structural submodules. Previous methodologies [10, 11] have either employed heuristics or developed learning pipelines to derive scores, achieving notable performance. Recently, the prevailingness of natural language prompts has facilitated an emerging wisdom that the success of AI is deeply rooted in the quality and specificity of data that is originally created by human [12]. Techniques such as in-context learning [13] and prompting [14] have been developed to create meticulously designed prompts or input templates to escalate the output quality

*These authors contributed equally.

Third Conference on Parsimony and Learning (CPAL 2026).

of LLMs. These strategies bolster the capabilities of LLMs and consistently achieve notable success across diverse downstream tasks. This offers a brand new angle for addressing the intricacies of structural pruning on importance estimation of vision models: *How can we leverage the potentials within the input space to facilitate the dissection of the relevance of each individual structural component across layers, thereby enhancing structural sparsity?*

One straightforward approach is directly editing input through visual prompt to enhance the performance of compressed vision models [15]. The performance upper bound of this approach largely hinges on the quality of the sparse model achieved by pruning, given that prompt learning is applied post-pruning. Moreover, when pruning is employed to address the intricate relevance between structural components across layers, the potential advantages of using visual prompts are not taken into consideration.

Therefore, we posit that probing judicious input editing is imperative for structural pruning to examine the importance of structural components in vision models. The **crux of our research** lies in embracing an innovative **data-centric** viewpoint towards structural pruning. Instead of designing or learning prompts on top of compressed models, we develop a novel end-to-end framework for channel pruning, which identifies and retains the most crucial channels across models by incorporating visual prompts, referred to as **PASS**.

Moreover, the complexities associated with inherent channel dependencies render the generation of sparse channel masks a challenging task. Due to this reason, many previous arts of pruning design delicate pruning metrics to recognize sparse subnetworks with smooth gradient flow [16]. To better handle the channel dependencies across layers during channel pruning, we propose to learn sparse masks using a **recurrent mechanism**. Specifically, the learned sparse mask for the recent layer largely depends on the mask from the previous layer in an efficient recurrent manner, and all the masks are learned by incorporating the extra information provided by visual prompts. The PASS framework is shown in Figure 1. Our contributions are summarized as follows:

- We probe and comprehend the role of the input editing in the context of channel pruning, and confirming the imperative to integrate visual prompts for crucial channel discovery.
- To handle the complex dependence caused by channel elimination across layers, we further develop a recurrent mechanism to efficiently learn layer-wise sparse masks by taking both the sparse masks from previous layers and visual prompts into consideration. Anchored by these innovations, we propose PASS, a pioneering framework dedicated to proficient channel pruning in convolution neural networks from a data-centric perspective.
- Through comprehensive evaluations across six datasets containing {CIFAR-10, CIFAR-100, Tiny-ImageNet, Food101, DTD, StanfordCars} and four architectures including {ResNet-18, ResNet-34, ResNet-50, VGG}, our results consistently demonstrate PASS's significant potential in enhancing both the performance of the resultant sparse models and computational efficiency.
- More interestingly, our empirical studies reveal that the sparse channel masks and the hypernetwork produced by PASS exhibit superior transferability, proving beneficial for a range of subsequent tasks.

2. Related Work

Structural Network Pruning. Structural pruning achieves network compression through entirely eliminating certain superfluous components from the dense network. In general, structural pruning follows three steps: (i) pre-training a large, dense model; (ii) pruning the unimportant channels based on criteria, and (iii) finetuning the pruned model to recover optimal performance. The primary contribution of various pruning approaches is located in the second step: proposing proper pruning metrics to identify the importance of channels. Some commonly-used pruning metric includes but not limited to weight norm [17], Taylor expansion [18], feature-maps reconstruction error [19] KL-divergence [20], greedy forward selection with largest loss reduction [21], feature-maps discriminant information [22].

Prompting. In the realm of natural language processing, prompting has been acknowledged as an effective strategy to adapt pre-trained models to specific tasks [23]. The power of this technique was highlighted by

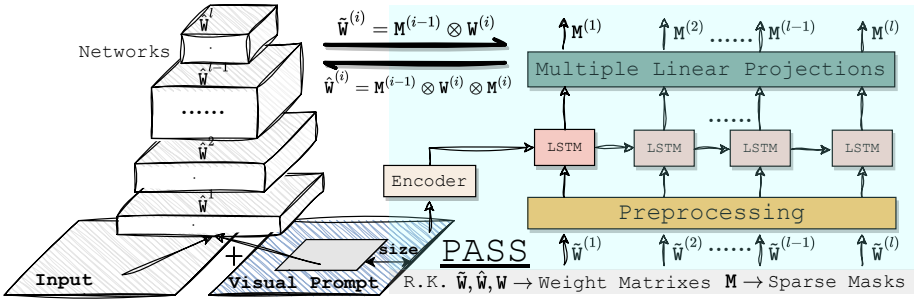

Figure 1: The overall framework of PASS. (*Left*) Our pruning target is a convolutional neural network (CNN) that takes images and visual prompts as input. (*Right*) The PASS hyper-network integrates the information from visual prompts and layer-wise weight statistics, then determines the significant structural topologies in a recurrent fashion.

GPT-3's successful generalization in transfer learning tasks using carefully curated text prompts [1]. Researchers have focused on refining text prompting methods [24] and developed a technique known as Prompt Tuning. This approach involves using prompts as task-specific continuous vectors optimized during fine-tuning [25], offering comparable performance to full fine-tuning with a significant reduction in parameter storage and optimization. Prompt tuning's application in the visual domain has seen significant advancement recently. Pioneered by [26], who introduced prompt parameters to input images, the concept was expanded by [27] to envelop input images with prompt parameters. [28] took this further, proposing visual prompt tuning for Vision Transformer models. Subsequently, [29] designed a prompt adapter to enhance these prompts. Concurrently, [30] integrated visual and text prompts in vision-language models, boosting downstream performance.

Hypernetwork. Hypernetworks represent a specialized form of network architecture, specifically designed to generate the weights of another Deep Neural Network (DNN). This design provides a meta-learning approach that enables dynamic weight generation and adaptability, which is crucial in scenarios where flexibility and learning efficiency are paramount. Initial iterations of hypernetworks, as proposed by [31], were configured to generate the weights for an entire target DNN. While this approach is favorable for smaller and less complex networks, it constrains the efficacy of hypernetworks when applied to larger and more intricate ones. To address this limitation, subsequent advancements in hypernetworks have been introduced, such as the component-wise generation of weights [32] and chunk-wise generation of weights [33]. Diverging from the initial goal of hypernetworks, our work employs them to fuse visual prompts and model information for generating sparse channel masks.

3. PASS: Visual Prompt Locates Good Structure Sparsity

Notations. Let us consider a CNN with l layers, and each layer i contains its corresponding weight tensor $\mathtt{W}^{(i)} \in \mathbb{R}^{\mathrm{C}_{\mathrm{O}}^i \times \mathrm{C}_{\mathrm{I}}^i \times \mathrm{K}^i \times \mathrm{K}^i}$, where $\{\mathrm{C}_{\mathrm{O}}^i, \mathrm{C}_{\mathrm{I}}^i, \text{and } \mathrm{K}^i\}$ are the number of output/input channels and convolutional kernel size, respectively. The entire parameter space for the network is defined as $\mathtt{W} = \{\mathtt{W}^{(i)}\}|_{i=1}^{l}$. Similarly, a layer-wise binary mask is represented by $\mathtt{M}^{(i)}$, where "0"/"1" indicates removing/maintaining the associated channel. $\mathtt{V}$ denotes our visual prompts. $(\boldsymbol{x}, y) \in \mathcal{D}$ denotes the data of a target task.

Rationale. In the realm of structural pruning for deep neural networks, one of the key challenges is how to derive channel-wise importance scores for each layer. Conventional mechanisms estimate the channel significance either in a global or layer-wise manner [10, 34], neglecting the sequential dependency between adjacency layers. Meanwhile, the majority of prevalent pruning methods are designed in a *model-centric* fashion [10]. In contrast, an ideal solution to infer the high-quality sparse mask for one neural network layer i should satisfy several conditions as follows:

① $\mathtt{M}^{(i)}$ *should be dependent to* $\mathtt{M}^{(i-1)}$. The sequential dependency between layers should be explicitly considered. It plays an essential role in encouraging gradient flow throughout the model [16, 35], by preserving structural "pathways".

② $\mathtt{M}^{(i)}$ *should be dependent to* $\mathtt{W}^{(i)}$. The statistics of network weights are commonly appreciated as powerful features for estimating channel importance [11, 17].

③ $\mathtt{M}^{(i)}$ *should be dependent to* $\mathtt{V}$. Motivated by the *data-centric* advances in NLP, such prompting can contribute to the dissecting and understanding of model behaviors [14, 36].

Therefore, it can be expressed as $\mathtt{M}^{(i)} = f(\mathtt{M}^{(i-1)}, \mathtt{W}^{(i)}, \mathtt{V})$, where the generation of a channel mask for layer i depends on the weights in the current layer, the previous layers' mask, and visual prompts.

3.1. Innovative Data-Model Co-designs through A Recurrent Hypernetwork

To meet the aforementioned requirements, PASS is proposed as illustrated in Figure 1, which enables the data-model co-design pruning via a recurrent hyper-network. Details are presented below.

Modeling the Layer Sequential Dependency. The recurrent hyper-network in PASS adopts a Long Short-Term Memory (LSTM) backbone since it is particularly suitable for capturing sequential dependency. It enables an "auto-regressive" way to infer the structural sparse mask. To be specific, the LSTM mainly utilizes the previous layer's mask $\mathtt{M}^{(i-1)}$, the current layer's weights $\mathtt{W}^{(i)}$, and a visual prompt $\mathtt{V}$ as follows:

$$\mathtt{M}^{(i)} = \mathtt{LSTM}_\theta(\widetilde{\mathtt{W}}^i, g_\omega(\mathtt{V}))\ \mathtt{M}^{(0)} = \mathtt{LSTM}(\mathtt{W}^{(i)}, g_\omega(\mathtt{V})), \tag{1}$$

where $\widetilde{\mathtt{W}}^{(i)} = \mathtt{M}^{(i-1)} \otimes \mathtt{W}^{(i)}$, the visual prompt $\mathtt{V}$ provides an initial hidden state for the LSTM hyper-network, θ is the parameters of the LSTM model, and $g_\omega(\mathtt{V})$ is the extra encoder to map the visual prompt into an embedding space. The channel-wise sparse masks ($\mathtt{M}^{(i)}$) generated from the hyper-network are utilized to prune the weights of each layer as expressed by $\widehat{\mathtt{W}}^{(i)} = \mathtt{M}^{(i-1)} \otimes \mathtt{W}^{(i)} \otimes \mathtt{M}^{(i)}$. $\mathtt{M}^{(i-1)} \otimes \mathtt{W}^{(i)}$ represents the pruning of in-channels while $\mathtt{W}^{(i)} \otimes \mathtt{M}^{(i)}$ denotes the pruning of out-channels.

Visual Prompt Encoder. An encoder is used to extract representations from the raw visual prompt $\mathtt{V}$. $g_\omega(\mathtt{V})$ denotes a three-layer convolution network and ω are the parameters for the CNN $g_\omega(\cdot)$. The dimension of extracted representations equals the dimension of the hidden state of the LSTM model. A learnable embedding will serve as the initial hidden state for the LSTM model.

Preprocessing the Weight. The in-channel pruned weights $\widetilde{\mathtt{W}}^{(i)}$ is a 4D matrix. In order to take this weight information, it is first transformed into a vector of length equal to the number of out-channels by averaging the weights over the $\mathrm{C}_\mathrm{I}^i \times \mathrm{K}^i \times \mathrm{K}^i$ dimensions. Then, these vectors are padded by zero elements to unify their length.

Converting Embedding to Channel-wise Sparse Mask. Generating layer-wise channel masks from the LSTM module presents two challenges: (1) it outputs embeddings of a uniform length, whereas the number of channels differs at each layer; (2) producing differentiable channel masks directly from this module is infeasible. To tackle these issues, PASS adopts a two-step approach: ① An independent linear layer is employed to map the learned embeddings onto channel-wise important scores corresponding to each layer. ② During the forward pass in training, the binary channel mask $\mathtt{M}$ is produced by setting the $(1-s)\times 100\%$ elements with the highest channel-wise important scores to 1, with the rest elements set to 0. In the backward pass, it is optimized by leveraging the straight-through estimation method [37]. Here the $s \in (0, 1)$ denotes the channel sparsity of the network layer.

For achieving an optimal non-uniform layer-wise sparsity ratio, we adopt global pruning [38] that eliminates the channels associated with the lowest score values from all layers during each optimization step. This approach is grounded in the findings of [10, 38], which demonstrate that layer-wise sparsity derived using this method surpasses other extensively researched sparsity ratios.

3.2. How to Optimize the Hypernetwork in PASS

Learning PASS. The procedures of learning PASS involves a jointly optimization of the visual prompt $\mathtt{V}$, encoder weights ω, and LSTM's model weights θ. Formally, it can be described below:

$$\min_{\theta,\omega,\mathtt{V}} \mathcal{L}(\Phi_{\widehat{\mathtt{W}}}(\boldsymbol{x}+\mathtt{V}), y),\ \widehat{\mathtt{W}}^{(i)} = \mathtt{M}^{(i-1)} \otimes \mathtt{W}^{(i)} \otimes \mathtt{M}^{(i)}, \tag{2}$$

Where $\Phi_{\widehat{\mathtt{W}}}(\cdot)$ is the target CNN with weights $\widehat{\mathtt{W}}$, $\boldsymbol{x}$ and y are the input image and its groundtruth label. Note that $\mathtt{M}^{(i)}$ is generated by $\mathtt{LSTM}_\theta(\widetilde{\mathtt{W}}^i, g_\omega(\mathtt{V}))$ as described in Equation 1. The objective of this learning phase

is to optimize the PASS model to generate layer-wise channel masks, leveraging both a visual prompt V and the inherent model weight statistics as guidance. After that, the obtained sparse subnetwork will be further fine-tuned on the downstream dataset.

Fine-tuning Sparse Subnetwork. The procedures of subnetwork fine-tuning involve the optimization of the visual prompt V and model weights W, which can be expressed by:

$$\min_{\mathtt{W},\mathtt{V}} \mathcal{L}(\Phi_{\widehat{\mathtt{W}}}(\boldsymbol{x}+\mathtt{V}), y), \tag{3}$$

where $\widehat{\mathtt{W}} = \mathtt{M}^{(i-1)} \otimes \mathtt{W}^{(i)} \otimes \mathtt{M}^{(i)}$ and the sparse channel mask M is fixed.

4. Experiments

In this section, we empirically demonstrate the effectiveness of our proposed PASS method against various baselines across multiple datasets and models. Additionally, we evaluate the transferability of the sparse channel masks and the hypernetwork learned by PASS. Further, we validate the superiority of our specific design by a series of ablations studies.

To evaluate PASS, we follow the widely-used evaluation of visual prompting which is pre-trained on large datasets and evaluated on various target domains [27, 28]. Specifically, this process is accomplished by two steps: (1) Identifying an optimal structural sparse neural network based on a pre-trained model and (2) Fine-tuning the structural sparse neural network on the target task. During the training process, we utilize the Frequency-based Label Mapping FLM as presented by [27] to facilitate the mapping of the logits from the pre-trained model to the logits of the target tasks.

4.1. Implementation Setups

Architectures and Datasets. We evaluate PASS using four traditional pre-trained models: ResNet-18, ResNet-34, ResNet-50, and VGG-16 without BatchNorm2D and three advanced models:ResNeXt-50,ViT-B/16 and Swin-T, all pre-trained on ImageNet-1K. Our evaluation contains six target tasks: Tiny-ImageNet, CIFAR-10/100, DTD, StanfordCars, and Food101.

Baselines. We select five popular structural pruning methods as our baselines: (1) *Group-L*1 *structural pruning* [9] reduces the network channels via l_1 regularization. (2) *GrowReg* [39] prunes the network channels via l_2 regularization with a growing penalty scheme. (3) *Slim* [11] imposes channel sparsity by applying l_1 regularization to the scaling factors in batch normalization layers. (4) *DepGraph* [10] models the inter-layer dependency and group-coupled parameters for pruning and (5) *ABC Pruner* [40] performs channel pruning through automatic structure search.

Training and Evaluation. We utilize off-the-shelf models from Torchvision [2] as the pre-trained models. During the pruning phase, we employ the SGD optimizer for the visual prompt, while the AdamW optimizer is used for the visual prompt encoder and the LSTM model for generating channel masks. Regarding the baselines, namely Group-L1 structural pruning, GrowReg, Slim, and DepGraph, they are trained based on this implementation [3] and ABC Prunner is trained based on their official public code [4]. During the fine-tuning phase, all pruned models, inclusive of those from PASS and the aforementioned baselines, are fine-tuned with the same hyper-parameters. For all experiments, we report the accuracy of the downstream task during testing and the floating point operations (FLOPs) for measuring the efficiency.

4.2. PASS Finds Good Structural Sparsity

In this section, we first validate the effectiveness of PASS across multiple downstream tasks and various model architectures. Subsequently, we investigate the transferability of both the generated channel masks and the associated model responsible for generating them.

[2] https://pytorch.org/vision/stable/index.html
[3] https://github.com/VainF/Torch-Pruning
[4] https://github.com/lmbxmu/ABCPruner

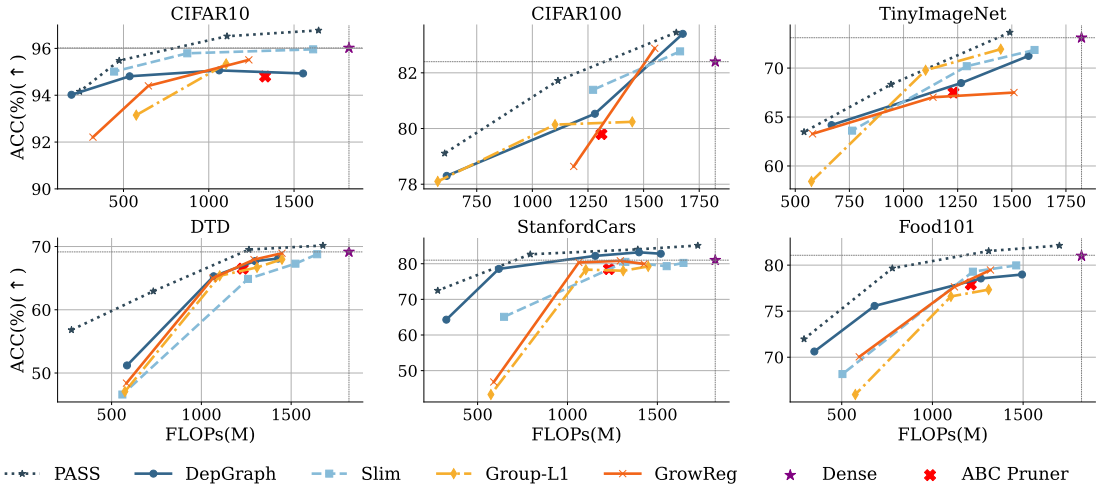

Figure 2: Test accuracy of channel-pruned networks across multiple downstream tasks based on the pre-trained ResNet-18.

Superior Performance across Downstream Tasks. In Figure 2, we present the test accuracy of the PASS method in comparison with several baseline techniques, including Group-L1, GrowReg, DepGraph, Slim, and ABC Prunner. The evaluation includes six downstream tasks: CIFAR-10, CIFAR-100, Tiny-ImageNet, DTD, StanfordCars, and Food101. The accuracies are reported against varying FLOPs to provide a comprehensive understanding of PASS's efficiency and performance.

From Figure 2, several salient observations can be drawn: ❶ PASS consistently demonstrates superior accuracy across varying FLOPs values for all six evaluated downstream tasks. On one hand, PASS achieves higher accuracy under the same FLOPs. For example, it achieves $1\% \sim 3\%$ higher accuracy than baselines under $1000\mathtt{M}$ FLOPs among all the datasets. On the other hand, PASS attains higher speedup[5] in achieving comparable accuracy levels. For instance, to reach accuracy levels of 96%, 81%, and 80% on CIFAR10, StanfordCars, and Food101 respectively, the PASS method consistently realizes a speedup of at least $0.35\times$ (900 VS 1400), outperforming the most competitive baseline. This consistent performance highlights the robustness and versatility of the PASS method across diverse scenarios. ❷ In terms of resilience to pruning, PASS exhibits a more gradual reduction in accuracy as FLOPs decrease. This trend is notably more favorable when compared with the sharper declines observed in other baseline methods. ❸ Remarkably, at the higher FLOPs levels, PASS not only attains peak accuracies but also surpasses the performance metrics of the fully fine-tuned dense models. For instance, PASS excels the fully fine-tuned dense models with $\{1.05\%, 0.99\%, 1.06\%\}$ on CIFRAR100, DTD and FOOD101 datasets.

Superior Performance across Model Architectures. We further evaluate the performance of PASS across multiple model architectures, namely VGG-16 without batch normalization [6], ResNet-34, and ResNet-50 and compare it with the baselines. The results are shown in Figure 3. We observe that our PASS achieves a competitive performance across all architectures, often achieving accuracy close to or even surpassing the dense models while being more computationally efficient. For instance, To achieve an accuracy of 75% on Tiny-ImageNet using ResNet-34/ResNet-50 and 66% accuracy using VGG-16, our PASS requires $0\% \sim 12\%$ fewer FLOPs compared to the most efficient baseline. These observations suggest that PASS can effectively generalize across different architectures, maintaining a balance between computational efficiency and model performance.

4.3. Experiments on ImageNet and Advanced Architectures

To draw a solid conclusion, we further conduct extensive experiments on a large dataset ImageNet using advanced pre-trained models such as ResNeXt-50, Swin-T, and ViT-B/16. The results are shown in Table 1. We observe that our method PASS demonstrates a significant speed-up with minimal accuracy loss, as indi-

[5]Following [10], we report the theoretical speedup ratios and it is defined as $\frac{\text{FLOPs}_{\texttt{PASS}} - \text{FLOPs}_{\texttt{baseline}}}{\text{FLOPs}_{\texttt{baseline}}}$

[6]The baseline Slim [11] is not applicable to this architecture.

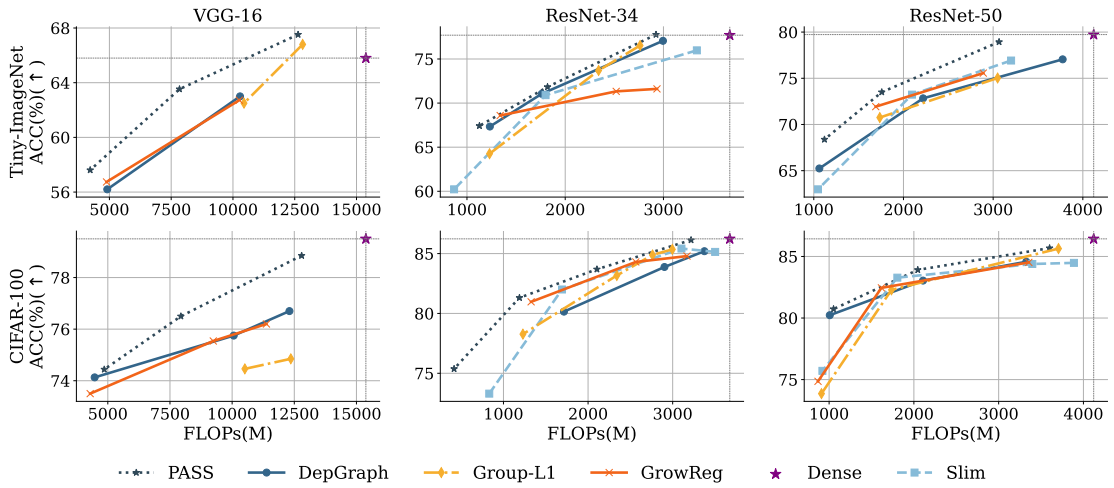

Figure 3: Test accuracy of channel-pruned networks across various architectures based on CIFAR-100 and Tiny-ImageNet.

cated by the Δ Acc., which is superior to existing methods like SSS [41], GFP [42], and DepGraph [10]. the resulting empirical evidence robustly affirms the effectiveness of PASS across both advanced neural network architectures and large-scale datasets.

4.4. Transferability of Learned Sparse Structure

Inspired by studies suggesting the transferability of subnetworks between tasks [43]. We investigate the transferability of PASS by posing two questions:(1) *Can the sparse channel masks, learned in one task, be effectively transferred to other tasks?* (2) *Is the hypernetwork, once trained, applicable to other tasks?* To answer Question (1), we test the accuracy of subnetworks found on Tiny-ImageNet when fine-tuning on CIFAR-10/100 and a pre-trained ResNet-18. To answer Question (2), we measure the accuracy of the subnetwork finetuning on the target datasets, i.e., CIFAR-10/100. This subnetwork is obtained by applying hypernetworks, trained on Tiny-ImageNet, to the visual prompts of the respective target tasks. The results are reported in Table 2. We observe that the channel mask and the hypernetwork, both learned by PASS, exhibit significant transferability on target datasets, highlighting their benefits across various subsequent tasks. More interestingly, the hypernetwork outperforms transferring the channel mask in most target tasks, providing two hints: ❶ Our learned hypernetworks can sufficiently capture the important topologies in downstream networks. Note that there is no parameter tuning for the hypernetworks and only with an adapted visual prompt. ❷ The visual prompt can effectively summarize the topological information from downstream neural networks, enabling superior sparsification.

Arch.	Method	Base	Pruned	Δ Acc.	FLOPs
ResNeXt-50	ResNeXt-50	77.62	-	-	4.27
	SSS [41]	77.57	74.98	-2.59	2.43
	GFP [42]	77.97	77.53	-0.44	2.11
	DepGraph [10]	77.62	76.48	-1.14	2.09
	Ours (PASS)	77.62	77.21	-0.41	2.01
ViT-B/16	ViT-B/16	81.07	-	-	17.6
	DepGraph [10]	81.07	79.17	-1.90	10.4
	Ours(PASS)	81.07	79.77	-1.30	10.7
Swin-T	Swin-T	81.4	-	-	4.49
	X-Pruner [44]	81.4	80.7	-0.7	3.2
	STEP [45]	81.4	77.2	-4.2	3.5
	Ours(PASS)	81.4	80.9	-0.5	3.4

Table 1: Pruning results based on ImageNet and Advanced models.

Channel Sparsity	10%		30%		50%	
	StanfordCars	CIFAR-100	StanfordCars	CIFAR-100	StanfordCars	CIFAR-100
DepGraph	75.79	81.60	69.26	76.90	45.10	69.40
Slim	58.10	80.27	43.00	71.86	26.3	68.56
Group-L1	76.50	79.80	58.30	72.60	20.40	58.50
Growreg	70.60	80.79	50.30	72.27	41.80	65.80
Transfer Channel Mask	83.50	82.45	79.70	80.83	76.60	78.81
Hypernetwork	84.31	82.49	79.88	80.98	76.80	78.67

Table 2: Transferability: Applying Channel Masks and Hypernetworks Learned from Tiny-ImageNet to CIFAR-100 and StanfordCars. The gray color denotes our method.

5. Ablations and Extra Invesitigations

Ablations on PASS. To evaluate the effectiveness of PASS, we pose two interesting questions about the design of its components: (1) *how do visual prompts and model weights contribute?* (2) *is the recurrent mechanism crucial for mask finding?* To answer the above questions, we conduct a series of ablation studies utilizing a pre-trained ResNet-18 on CIFAR-100. The extensive investigations contain (1) *dropping either the visual prompt or model weights*; (2) *destroy the recurrent nature in our hypernetwork*, such as using a Convolutional Neural Network (CNN) or a Multilayer Perceptron (MLP) to replace LSTM. The results are collected in Table 3. We observe that ❶ The exclusion of either the visual prompt or model weights leads to a pronounced drop in test accuracy (e.g., $83.45\% \rightarrow 82.83\%$ and 82.66% respectively at 90% channel density), indicating the essential interplay role of both visual prompt and model weights in sparsification. ❷ If the recurrent nature in our design is destroyed, *i.e.*, MLP or CNN methods variants, it suffers a performance decrement (*e.g.*, $81.72\% \rightarrow 81.07\%$ and 81.09% respectively at 70% channel density). It implies a Markov property during the sparsification of two adjacent layers, which echoes the sparsity pathway findings in [16].

Channel Sparsity =		10%	30%	50%	70%
Input Ablations	LSTM+VP	82.66	81.20	77.94	72.01
	LSTM+Weights	82.83	81.13	77.83	72.45
	LSTM+Weights+VP(Ours)	83.45	81.72	79.11	73.53
Architecture Ablations	ConVNet+VP	83.21	81.09	78.15	72.31
	MLP+VP+Weights	83.23	81.07	77.84	72.38
	LSTM+Weights+VP(Ours)	83.45	81.72	79.11	73.53

Table 3: Ablations for PASS based on CIFAR-100 using a pre-trained ResNet-18.

Ablations on Visual Prompt. A visual prompt is a patch integrated with the input, as depicted in Figure 1. Two prevalent methods for incorporating the visual prompt into the input have been identified in the literature [26, 27]:(1) Adding to the input (abbreviated as **"Additive visual prompt"**. (2)Expanding around the perimeter of the input, namely, the input is embedded into the central hollow section of the visual prompt (abbreviated as **"Expansive visual prompt"**). As discussed in section 5, visual prompt (VP) plays a key role in PASS. Therefore, we pose such a question:*How do the strategies and size of VP influence the performance of PASS?* To address this concern, we conduct experiments with "Additive visual prompt" and "Expansive visual prompt" respectively on CIFAR-100 using a pre-trained ResNet-18 under 10%, 30% and 50% channel sparsities, and we also show the performance of PASS with varying the VP size from 0 to 48. The results are shown in Figure 4. We conclude that ❶ "Additive visual prompt" performs better than "Expansive visual prompt" across different sparsities. The disparity might be from the fact that "Expansive visual prompt" requires resizing the input to a smaller dimension, potentially leading to information loss, a problem that "Additive visual prompt" does not face. ❷ The size of VP impacts the performance of PASS. We observe that test accuracy initially rises with the increase in VP size but starts to decline after reaching a peak at size 16. A potential explanation for this decline is that the larger additive VP might overlap a significant portion of the input, leading to the loss of crucial information.

Impact of Hidden Size in HyperNetwork It is well-known that model size is an important factor impacting its performance, inducing the question *how does the size of hypernetwork influence the performance of PASS?*.

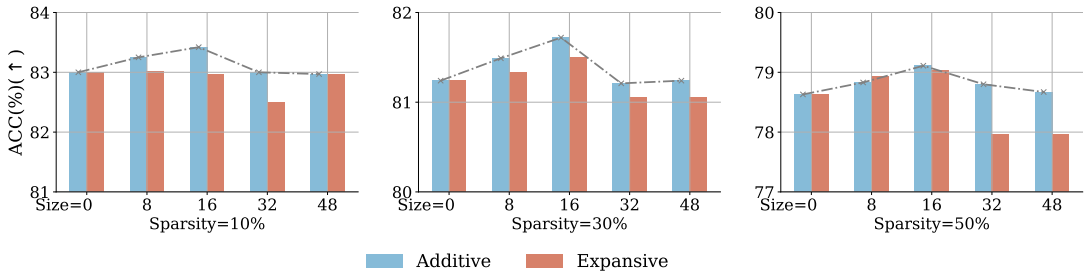

Figure 4: Ablation study on visual prompt strategies and their sizes. Experiments are conducted on CIFAR-100 and a pre-trained ResNet-18.

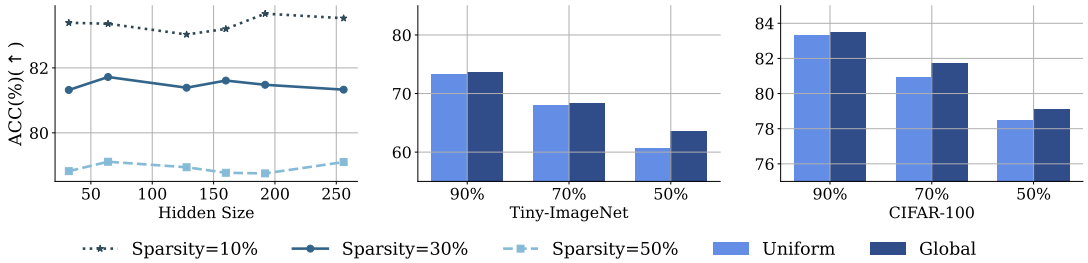

Figure 5: (1)Ablation study of the hypernetwork's hidden size (Left Figure) using a pre-trained ResNet-18 on CIFAR-100. (2)Comparison between Global Pruning and Uniform Pruning strategies (Middle and Right Figures) using a pre-trained ResNet-18 on CIFAR-100 and Tiny-Imagenet.

To address this concern, we explore the impact of the hypernetwork hidden sizes on PASS by varying the hidden size of the proposed hypernetwork from 32 to 256 and evaluate its performance on CIFAR100 using a pre-trained ResNet-18 model under 10%, 30% and 50% channel sparsity respectively. The results are presented in the left figure of Figure 5. We observe that the hidden size doesn't drastically affect the accuracy. While there are fluctuations, they are within a small range, suggesting that the hidden size is not a dominant factor in influencing the performance of PASS.

Uniform Pruning VS Global Pruning. When converting the channel-wise importance scores into the channel masks, there are two prevalent strategies: (1) *Uniform Pruning*. [38, 46] It prunes the channels of each layer with the lowest important scores by the same proportion. (2) *Global Pruning*. [10, 38] It prunes channels with the lowest important scores from all layers, leading to varied sparsity across layers. In this section, we evaluate the performance of global pruning and uniform pruning for PASS on CIFAR-100 using a pre-trained ResNet-18, with results presented in Figure 5. We observe that global pruning consistently yields higher test accuracy than uniform pruning, indicating its superior suitability for PASS, also reconfirming the importance of layer sparsity in sparsifying neural networks [38, 47].

6. Conclusion

In this paper, we delve deep into structural model pruning, with a particular focus on leveraging the potential of visual prompts for discerning channel importance in vision models. Our exploration highlights the key role of the input space and how judicious input editing can significantly influence the efficacy of structural pruning. We propose PASS, an innovative, end-to-end framework that harmoniously integrates visual prompts, providing a data-centric lens to channel pruning. Our recurrent mechanism adeptly addressed the intricate channel dependencies across layers, ensuring the derivation of high-quality structural sparsity.

Extensive evaluations across six datasets and four architectures underscore the prowess of PASS. The PASS framework excels not only in performance and computational efficiency but also demonstrates that its pruned models possess notable transferability. In essence, this research paves a new path for channel pruning, underscoring the importance of intertwining data-centric approaches with traditional model-centric methodologies. The fusion of these paradigms, as demonstrated by our findings, holds immense promise for the future of efficient neural network design.

Acknowledgements

The authors acknowledge the use of resources provided by the Isambard-AI National AI Research Resource (AIRR) and the Dutch national e-infrastructure, supported by the SURF Cooperative (Project EINF-17091). Isambard-AI is operated by the University of Bristol and funded by the UK Government's Department for Science, Innovation and Technology (DSIT) via UK Research and Innovation and the Science and Technology Facilities Council [ST/AIRR/I-A-I/1023]. Finally, we thank the anonymous reviewers for their insightful comments, which significantly improved the quality of this paper.

References

[1] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

[2] Wei-Lin Chiang, Zhuohan Li, Zi Lin, Ying Sheng, Zhanghao Wu, Hao Zhang, Lianmin Zheng, Siyuan Zhuang, Yonghao Zhuang, Joseph E Gonzalez, et al. Vicuna: An open-source chatbot impressing gpt-4 with 90%* chatgpt quality. *Arxiv preprint*, 2023.

[3] Mostafa Dehghani, Josip Djolonga, Basil Mustafa, Piotr Padlewski, Jonathan Heek, Justin Gilmer, Andreas Peter Steiner, Mathilde Caron, Robert Geirhos, Ibrahim Alabdulmohsin, et al. Scaling vision transformers to 22 billion parameters. In *International Conference on Machine Learning*, pages 7480–7512. PMLR, 2023.

[4] Jinze Bai, Shuai Bai, Shusheng Yang, Shijie Wang, Sinan Tan, Peng Wang, Junyang Lin, Chang Zhou, and Jingren Zhou. Qwen-vl: A frontier large vision-language model with versatile abilities. *arXiv preprint arXiv:2308.12966*, 2023.

[5] Tim Dettmers, Mike Lewis, Younes Belkada, and Luke Zettlemoyer. Llm. int8 (): 8-bit matrix multiplication for transformers at scale. *arXiv preprint arXiv:2208.07339*, 2022.

[6] Guangxuan Xiao, Ji Lin, Mickael Seznec, Hao Wu, Julien Demouth, and Song Han. Smoothquant: Accurate and efficient post-training quantization for large language models. In *International Conference on Machine Learning*, pages 38087–38099. PMLR, 2023.

[7] Tianjin Huang, Lu Yin, Zhenyu Zhang, Li Shen, Meng Fang, Mykola Pechenizkiy, Zhangyang Wang, and Shiwei Liu. Are large kernels better teachers than transformers for convnets? *ICML*, 2023.

[8] Siqi Sun, Yu Cheng, Zhe Gan, and Jingjing Liu. Patient knowledge distillation for bert model compression. *arXiv preprint arXiv:1908.09355*, 2019.

[9] Hao Li, Asim Kadav, Igor Durdanovic, Hanan Samet, and Hans Peter Graf. Pruning filters for efficient convnets. In *International Conference on Learning Representations*, 2017. URL `https://openreview.net/forum?id=rJqFGTslg`.

[10] Gongfan Fang, Xinyin Ma, Mingli Song, Michael Bi Mi, and Xinchao Wang. Depgraph: Towards any structural pruning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16091–16101, 2023.

[11] Zhuang Liu, Jianguo Li, Zhiqiang Shen, Gao Huang, Shoumeng Yan, and Changshui Zhang. Learning efficient convolutional networks through network slimming. In *Proceedings of the IEEE international conference on computer vision*, pages 2736–2744, 2017.

[12] Daochen Zha, Zaid Pervaiz Bhat, Kwei-Herng Lai, Fan Yang, Zhimeng Jiang, Shaochen Zhong, and Xia Hu. Data-centric artificial intelligence: A survey. *arXiv preprint arXiv:2303.10158*, 2023.

[13] Mingda Chen, Jingfei Du, Ramakanth Pasunuru, Todor Mihaylov, Srini Iyer, Veselin Stoyanov, and Zornitsa Kozareva. Improving in-context few-shot learning via self-supervised training. *arXiv preprint arXiv:2205.01703*, 2022.

[14] Anastasia Razdaibiedina, Yuning Mao, Rui Hou, Madian Khabsa, Mike Lewis, and Amjad Almahairi. Progressive prompts: Continual learning for language models. *arXiv preprint arXiv:2301.12314*, 2023.

[15] Zhaozhuo Xu, Zirui Liu, Beidi Chen, Yuxin Tang, Jue Wang, Kaixiong Zhou, Xia Hu, and Anshumali Shrivastava. Compress, then prompt: Improving accuracy-efficiency trade-off of llm inference with transferable prompt. *arXiv preprint arXiv:2305.11186*, 2023.

[16] Chaoqi Wang, Guodong Zhang, and Roger Grosse. Picking winning tickets before training by preserving gradient flow. *arXiv preprint arXiv:2002.07376*, 2020.

[17] Hao Li, Asim Kadav, Igor Durdanovic, Hanan Samet, and Hans Peter Graf. Pruning filters for efficient convnets. *arXiv preprint arXiv:1608.08710*, 2016.

[18] Pavlo Molchanov, Stephen Tyree, Tero Karras, Timo Aila, and Jan Kautz. Pruning convolutional neural networks for resource efficient inference. *arXiv preprint arXiv:1611.06440*, 2016.

[19] Yihui He, Ji Lin, Zhijian Liu, Hanrui Wang, Li-Jia Li, and Song Han. Amc: Automl for model compression and acceleration on mobile devices. In *Proceedings of the European conference on computer vision (ECCV)*, pages 784–800, 2018.

[20] Jian-Hao Luo and Jianxin Wu. Neural network pruning with residual-connections and limited-data. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1458–1467, 2020.

[21] Mao Ye, Chengyue Gong, Lizhen Nie, Denny Zhou, Adam Klivans, and Qiang Liu. Good subnetworks provably exist: Pruning via greedy forward selection. *Proceedings of International Conference on Machine Learning*, pages 10820–10830, 2020.

[22] Zejiang Hou and Sun-Yuan Kung. A feature-map discriminant perspective for pruning deep neural networks. *arXiv preprint arXiv:2005.13796*, 2020.

[23] Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. *ACM Computing Surveys*, 55(9):1–35, 2023.

[24] Taylor Shin, Yasaman Razeghi, Robert L Logan IV, Eric Wallace, and Sameer Singh. Autoprompt: Eliciting knowledge from language models with automatically generated prompts. *arXiv preprint arXiv:2010.15980*, 2020.

[25] Xiang Lisa Li and Percy Liang. Prefix-tuning: Optimizing continuous prompts for generation. *arXiv preprint arXiv:2101.00190*, 2021.

[26] Hyojin Bahng, Ali Jahanian, Swami Sankaranarayanan, and Phillip Isola. Exploring visual prompts for adapting large-scale models. *arXiv preprint arXiv:2203.17274*, 1(3):4, 2022.

[27] Aochuan Chen, Yuguang Yao, Pin-Yu Chen, Yihua Zhang, and Sijia Liu. Understanding and improving visual prompting: A label-mapping perspective. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 19133–19143, 2023.

[28] Menglin Jia, Luming Tang, Bor-Chun Chen, Claire Cardie, Serge Belongie, Bharath Hariharan, and Ser-Nam Lim. Visual prompt tuning. In *European Conference on Computer Vision*, pages 709–727. Springer, 2022.

[29] Weihuang Liu, Xi Shen, Chi-Man Pun, and Xiaodong Cun. Explicit visual prompting for low-level structure segmentations. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 19434–19445, 2023.

[30] Yuhang Zang, Wei Li, Kaiyang Zhou, Chen Huang, and Chen Change Loy. Unified vision and language prompt learning. *arXiv preprint arXiv:2210.07225*, 2022.

[31] Chris Zhang, Mengye Ren, and Raquel Urtasun. Graph hypernetworks for neural architecture search. *arXiv preprint arXiv:1810.05749*, 2018.

[32] Dominic Zhao, Seijin Kobayashi, João Sacramento, and Johannes von Oswald. Meta-learning via hypernetworks. In *4th Workshop on Meta-Learning at NeurIPS 2020 (MetaLearn 2020)*. NeurIPS, 2020.

[33] Vinod Kumar Chauhan, Jiandong Zhou, Soheila Molaei, Ghadeer Ghosheh, and David A Clifton. Dynamic inter-treatment information sharing for heterogeneous treatment effects estimation. *arXiv preprint arXiv:2305.15984*, 2023.

[34] Yihui He, Xiangyu Zhang, and Jian Sun. Channel pruning for accelerating very deep neural networks. *Proceedings of the IEEE International Conference on Computer Vision*, pages 1389–1397, 2017.

[35] Hoang Pham, Anh Ta, Shiwei Liu, Dung D Le, and Long Tran-Thanh. Understanding pruning at initialization: An effective node-path balancing perspective. 2022.

[36] Xiang Chen, Ningyu Zhang, Xin Xie, Shumin Deng, Yunzhi Yao, Chuanqi Tan, Fei Huang, Luo Si, and Huajun Chen. Knowprompt: Knowledge-aware prompt-tuning with synergistic optimization for relation extraction. In *Proceedings of the ACM Web conference 2022*, pages 2778–2788, 2022.

[37] Yoshua Bengio, Nicholas Léonard, and Aaron Courville. Estimating or propagating gradients through stochastic neurons for conditional computation. *arXiv preprint arXiv:1308.3432*, 2013.

[38] Tianjin Huang, Tianlong Chen, Meng Fang, Vlado Menkovski, Jiaxu Zhao, Lu Yin, Yulong Pei, Decebal Constantin Mocanu, Zhangyang Wang, Mykola Pechenizkiy, et al. You can have better graph neural networks by not training weights at all: Finding untrained gnns tickets. *arXiv preprint arXiv:2211.15335*, 2022.

[39] Huan Wang, Can Qin, Yulun Zhang, and Yun Fu. Neural pruning via growing regularization. In *International Conference on Learning Representations*, 2021. URL `https://openreview.net/forum?id=o966_Is_nPA`.

[40] Mingbao Lin, Rongrong Ji, Yuxin Zhang, Baochang Zhang, Yongjian Wu, and Yonghong Tian. Channel pruning via automatic structure search. *arXiv preprint arXiv:2001.08565*, 2020.

[41] Zehao Huang and Naiyan Wang. Data-driven sparse structure selection for deep neural networks. In *Proceedings of the European conference on computer vision (ECCV)*, pages 304–320, 2018.

[42] Liyang Liu, Shilong Zhang, Zhanghui Kuang, Aojun Zhou, Jing-Hao Xue, Xinjiang Wang, Yimin Chen, Wenming Yang, Qingmin Liao, and Wayne Zhang. Group fisher pruning for practical network compression. In *International Conference on Machine Learning*, pages 7021–7032. PMLR, 2021.

[43] Tianlong Chen, Jonathan Frankle, Shiyu Chang, Sijia Liu, Yang Zhang, Zhangyang Wang, and Michael Carbin. The lottery ticket hypothesis for pre-trained bert networks. *Advances in neural information processing systems*, 33:15834–15846, 2020.

[44] Lu Yu and Wei Xiang. X-pruner: explainable pruning for vision transformers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 24355–24363, 2023.

[45] Jiaoda Li, Ryan Cotterell, and Mrinmaya Sachan. Differentiable subset pruning of transformer heads. *Transactions of the Association for Computational Linguistics*, 9:1442–1459, 2021.

[46] Vivek Ramanujan, Mitchell Wortsman, Aniruddha Kembhavi, Ali Farhadi, and Mohammad Rastegari. What's hidden in a randomly weighted neural network? In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11893–11902, 2020.

[47] Shiwei Liu, Tianlong Chen, Xiaohan Chen, Li Shen, Decebal Constantin Mocanu, Zhangyang Wang, and Mykola Pechenizkiy. The unreasonable effectiveness of random pruning: Return of the most naive baseline for sparse training. In *International Conference on Learning Representations*, 2022. URL `https://openreview.net/forum?id=VBZJ_3tz-t`.

[48] Yizeng Han, Gao Huang, Shiji Song, Le Yang, Honghui Wang, and Yulin Wang. Dynamic neural networks: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(11):7436–7456, 2021.

A. Parameters of Hypernetworks

In this study, the hidden size of the hypernetwork is configured to 64. A detailed breakdown of the number of parameters for the hypernetworks utilized in this research is provided in Table 4. It is noteworthy that the parameter count for the hypernetworks is significantly lower compared to that of the pretrained models. For instance, in the case of ResNet-18, the hypernetwork parameters constitute only 2.8% of the total parameters of the pre-trained ResNet-18.

Table 4: The number of parameters for our Hypernetworks.

	ResNet-18 (11M)	ResNet-34 (21M)	ResNet-50 (25M)	VGG-16 (138M)
#Parameters-HyperNetwork	0.31M (2.8%)	0.56M (2.6%)	1.5M (6%)	0.34M (0.2%)

B. Implementation Details

Table 5 summarizes the hyper-parameters for `PASS` used in all our experiments.

Table 5: Implementation details on each dataset.

Settings	Tny-ImageNet	CIFAR-10	CIFAF-10	DTD	StanfordCars	Food101
Stage 1: Learning to Prune						
Batch Size	128					
Weight Decay - VP	0	0	0	0	0	0
Learning Rate - VP	$1e-2$	$1e-2$	$1e-2$	$1e-2$	$1e-2$	$1e-2$
Optimizer - VP	SGD optimizer					
LR-Decay-Scheduler - VP	cosine					
Weight Decay - HyperNetwork	$1e-2$	$1e-2$	$1e-2$	$1e-2$	$1e-2$	$1e-2$
Learning Rate - HyperNetwork	$1e-3$	$1e-3$	$1e-3$	$1e-3$	$1e-3$	$1e-3$
Optimizer - HyperNetwork	AdamW optimizer					
LR-Decay-Scheduler - HyperNetwork	cosine					
Total epochs	50					
Stage 2: Fine-tune						
Batch Size	128					
Weight Decay - VP	0	0	0	0	0	0
Learning Rate - VP	$1e-3$	$1e-2$	$1e-2$	$1e-2$	$1e-2$	$1e-2$
Optimizer - VP	SGD optimizer					
LR-Decay-Scheduler - VP	cosine					
Weight Decay - Pruned Network	$5e-4$	$3e-4$	$5e-4$	$5e-4$	$5e-4$	$5e-4$
Learning Rate - Pruned Network	$1e-3$	$1e-2$	$1e-2$	$1e-2$	$1e-2$	$1e-2$
Optimizer - Pruned Network	SGD optimizer					
LR-Decay-Scheduler - Pruned Network	multistep-$\{6, 8\}$	cosine	cosine	cosine	cosine	cosine
Total epochs	10	50	50	50	50	50

C. Learned Channel Sparsity

We present the channel sparsity learned by `PASS` on CIFAR-100 and Tiny-ImageNet using a pre-trained ResNet-18 in Table 6. Our observations indicate that channel sparsity is generally higher in the top layers and lower in the bottom layers of the network.

D. Complexity Analysis of the Hypernetwork

In this section, we provide a comprehensive analysis about the complexity of the Hypernetwork. (1) Regarding the impact on time complexity, our recurrent hyper-network is designed for efficiency. The channel masks are pre-calculated, eliminating the need for real-time generation during both the inference and subnetwork fine-tuning phases. Therefore, the recurrent hyper-network does not introduce any extra time complexity during the inference and the fintune-tuning phase. The additional computing time is limited to the phase of channel mask identification. (2) Moreover, the hyper-network itself is designed to be lightweight. The

Table 6: Layer-wise sparsity of the pre-trained ResNet-18 on CIFAR-100 and Tiny-ImageNet as learned by PASS at $30\%, 50\%$ sparsity levels.

Layer	Fully Dense #Channels	CIFAR-100		Tiny-ImageNet	
		30% Sparsity	50% Sparsity	30% Sparsity	50% Sparsity
Layer 1 - conv1	64	9.4%	29.7%	20.3%	37.5%
Layer 2 - layer1.0.conv1	64	17.2%	43.8%	28.1%	56.2%
Layer 3 - layer1.0.conv2	64	29.7%	29.7%	20.3%	39.0%
Layer 4 - layer1.1.conv1	64	15.7%	46.9%	50%	62.5%
Layer 5 - layer1.1.conv2	64	22.7%	26.6%	17.1%	32.8%
Layer 6 - layer2.0.conv1	128	19.6%	46.9%	42.9%	57.0%
Layer 7 - layer2.0.conv2	128	19.6%	45.4%	14.0%	34.3%
Layer 8 - layer2.0.downsample.0	128	19.6%	45.4%	14.0%	34.3%
Layer 9 - layer2.1.conv1	128	19.6%	44.6%	34.3%	55.5%
Layer 10 - layer2.1.conv2	128	7.1%	28.9%	16.4%	34.3%
Layer 11 - layer3.0.conv1	256	29%	50%	42.1%	58.9%
Layer 12 - layer3.0.conv2	256	15.3%	50%	11.7%	28.5%
Layer 13 - layer3.0.downsample.0	256	15.3%	50%	11.7%	28.5%
Layer 14 - layer3.1.conv1	256	27.4%	43.8%	33.2%	52.3%
Layer 15 - layer3.1.conv2	256	11%	26.2%	10.1%	23.8%
Layer 16 - layer4.0.conv1	512	29.3%	50%	33.3%	54.4%
Layer 17 - layer4.0.conv2	512	41.8%	50%	36.1%	58.9%
Layer 18 - layer4.0.downsample.0	512	41.8%	50%	36.1%	58.9%
Layer 19 - layer4.1.conv1	512	44.6%	48.3%	39.2%	65.8%
Layer 20 - layer4.1.conv2	512	42.6%	49.9%	27.1%	46.2%
Layer 21 - Linear	512	0%	0%	0%	0%

number of parameters it contributes to the overall model is minimal, thus ensuring that any additional complexity during the mask-finding phase is negligible. This claim is substantiated by empirical observations: the hyper-network accounts for only about 0.2% to 6% of the total model parameters across various architectures such as ResNet-18/34 and VGG-16, as illustrated in Table 4. (3) Additionally, we assessed the training time per epoch with and without the hyper-network during the channel mask identification phase. Our findings in Table 7 indicate that the inclusion of the LSTM network has a marginal effect on these durations, further affirming the efficiency of our approach.

Table 7: Per-Epoch Training Time (s) With/Without Hypernetworks in Channel Mask Identification (1× A100)

	ResNet-18 (11M)	ResNet-34 (21M)	ResNet-50 (25M)
w/o HyperNetwork	70.05	73.95	95.65
w/ HyperNetwork	72.2	76.95	111.6

E. Difference between our proposed PASS and dynamic neural network

There are two fundamental differences between our proposed PASS and dynamic neural network. (1) **The hyper-network in our proposed PASS is not 'dynamic'.** Dynamic neural networks, as categorized in the literature, are networks capable of adapting their structures or parameters conditioned in a sample-dependent manner, as outlined in [48]. In contrast, the hyper-network within our PASS framework does not exhibit this 'dynamic' nature. It is designed to be dependent on a visual prompt (task-specific), as opposed to dynamically adjusting to input samples. This hyper-network's role is confined to the channel mask identification phase and is not employed during the inference phase. Therefore, it is fundamentally different from dynamic neural networks. (2) **Their goals are different.** The fundamental goal of the hyper-network in PASS is distinct from that of dynamic neural networks. While the latter focuses on adapting their architecture or parameters based on input samples, our hyper-network is specifically engineered for the integration of visual prompts with the statics of model weights.