Pruning-based Data Selection and Network Fusion for Efficient Deep Learning

Humaira Kousar* **Hasnain Irshad Bhatti*** **Jaekyun Moon**

KAIST

Abstract

Efficient data selection is essential for improving the training efficiency of deep neural networks and reducing the associated annotation costs. However, traditional methods tend to be computationally expensive, limiting their scalability and real-world applicability. We introduce PruneFuse, a novel method that combines pruning and network fusion to enhance data selection and accelerate network training. In PruneFuse, the original dense network is pruned to generate a smaller surrogate model that efficiently selects the most informative samples from the dataset. Once this iterative data selection selects sufficient samples, the insights learned from the pruned model are seamlessly integrated with the dense model through network fusion, providing an optimized initialization that accelerates training. Extensive experimentation on various datasets demonstrates that PruneFuse significantly reduces computational costs for data selection, achieves better performance than baselines, and accelerates the overall training process.

1 Introduction

Deep learning models have achieved remarkable success across various domains, ranging from image recognition to natural language processing [1–3]. However, the performance of models heavily relies on the access of large amounts of labeled data for training [4]. In real-world applications, manually annotating massive datasets can be prohibitively expensive and time-consuming. Data selection techniques such as Active Learning (AL) [5] offer a promising solution to address this challenge by iteratively selecting the most informative samples from the unlabeled dataset for annotation. The goal of AL is to reduce labeling costs while maintaining or improving model performance. However, as data and modal complexity grow, traditional AL techniques that require iterative model training become computationally expensive, limiting scalability in resource-constrained settings.

In this paper, we propose PruneFuse, a novel strategy for efficient data selection in active learning setting that overcomes the limitations of traditional approaches. Our approach is based on model pruning, which reduces the complexity of neural networks. By utilizing small pruned networks for data selection, we eliminate the need to train large models during the data selection phase, thus significantly reducing computational demands. Additionally after the data selection phase, we utilize the learning of these pruned networks to train the final model through a fusion process, which harnesses the insights from the trained networks to accelerate convergence and improve the generalization of the final model.

Contributions. Our key contribution is to introduce an efficient and rapid data selection technique that leverages pruned networks. By employing pruned networks as data selectors, PruneFuse ensures computationally efficient selection of informative samples leading to overall superior generalization.

*Equal Contribution. Correspondence to: Humaira Kousar <humairakousar32@kaist.ac.kr>
Department of Electrical Engineering, KAIST

38th Conference on Neural Information Processing Systems (NeurIPS 2024).

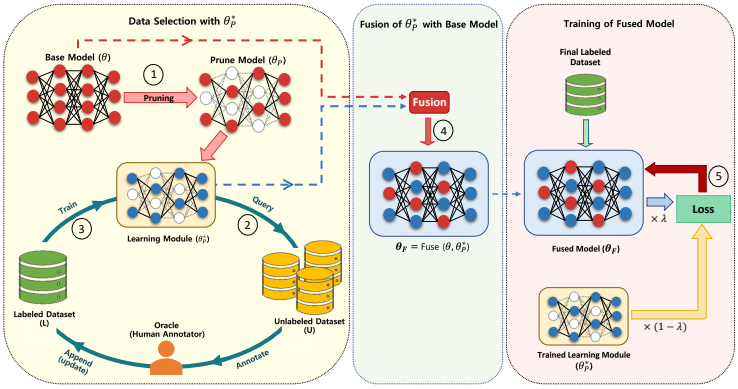


Figure 1: **Overview of the PruneFuse Method**: (1) An untrained neural network is initially pruned to form a structured, pruned network θ_p. (2) This pruned network θ_p queries the dataset to select prime candidates for annotation, similar to active learning techniques. (3) θ_p is then trained on these labeled samples to form the trained pruned network θ_p^*. (4) The trained pruned network θ_p^* is fused with the base model θ, resulting in a fused model. (5) The fused model is further trained on a selected subset of the data, incorporating knowledge distillation from θ_p^*.

Furthermore, we propose a novel concept of fusing these pruned networks with the original untrained model, enhancing model initialization and accelerating convergence during training.

We demonstrate the broad applicability of PruneFuse across various network architectures, offering a flexible tool for efficient data selection in diverse deep learning settings. Extensive experimentation on CIFAR-10, CIFAR-100, and Tiny-ImageNet-200 shows that PruneFuse achieves superior performance to state-of-the-art active learning methods while significantly reducing computational costs.

2 Background and Related Works

Subset Selection Framework. Active Learning (AL) is widely utilized iterative approach tailored for situations with abundant unlabeled data. Given a classification task with C classes and a large pool of unlabeled samples U, AL revolves around selectively querying the most informative samples from U for labeling. The process commences with an initial set of randomly sampled data s^0 from U, which is subsequently labeled. In subsequent rounds, AL augments the labeled set L by adding newly identified informative samples. This cycle repeats until a predefined number of labeled samples b are selected.

Data Selection. Approaches such as [6, 7, 5] aim to select informative samples using techniques like diversity maximization and Bayesian uncertainty estimation. Parallelly, the domain of active learning has unveiled strategies, such as [8, 9, 7, 10, 11, 6], which prioritize samples that can maximize information gain, thereby enhancing model performance with minimal labeling effort. While these methods achieve efficient data selection, they still require training large models for the selection process, resulting in significant computational overhead. Other strategies such as [12] optimize this selection process by matching the gradients of subset with training or validation set based on orthogonal matching algorithm and [13] performs meta-learning based approach for online data selection. SubSelNet [14] proposes to approximate a model that can be used to select the subset for various architectures without retraining the target model, hence reducing the overall overhead. However, it involves pre-training routine which is very costly and needed again for any change in data or model distribution. SVP [15] introduces to use small proxy models for data selection but discards these proxies before training the target model. Additionally, structural discrepancies between the proxy and target models may result in sub-optimal data selections. Our approach also builds on this foundation of using small model (which in our case is a pruned model) but it enables direct integration with the target model through the fusion process. This ensures that the knowledge acquired during data selection is retained and actively contributes to the training of the original model. Also, the architectural coherence between the pruned and the target model provides a more seamless and effective mechanism for data selection, enhancing overall model performance and efficiency.

Efficient Deep Learning. Methods such as [16–23] have been proposed to reduce model size and computational requirements. Neural Network pruning has been extensively investigated as a technique to reduce the complexity of deep neural networks [18]. Pruning strategies can be broadly divided into Unstructured Pruning [24–27] and Structured Pruning [28–31] based on the granularity

and regularity of the pruning scheme. Unstructured pruning often yields a superior accuracy-size trade-off whereas structured pruning offers practical speedup and compression without necessitating specialized hardware [32]. While pruning literature suggests pruning after training [33] or during training [34, 35], recent research explore the viability of pruning at initialization [36–38, 37, 39]. In our work, we leverage the benefits of pruning at initialization to create a small representative model for efficient data selection.

3 PruneFuse

In this section, we delineate the PruneFuse methodology, illustrated in Fig. 1 (and Algorithm 1 provided in Appendix). The procedure begins with network pruning at initialization, offering a streamlined model for data selection. Upon attaining the desired data subset, the pruned model undergoes a fusion process with the original network, leveraging the structural coherence between them. The fused model is subsequently refined through knowledge distillation, enhancing its performance. We framed the problem as, let s_p be the subset selected using a pruned model θ_p and s be the subset selected using the original model θ. We want to minimize:

$$\arg\min_{s_p} \left| E_{(x,y)\in s_p}[l(x,y;\theta,\theta_p)] - E_{(x,y)\in D}[l(x,y;\theta)] \right| \tag{1}$$

Where $E_{(x,y)\in s_p}[l(x,y;\theta,\theta_p)]$ is the expected loss on subset s_p (selected using θ_p) when evaluated using the original model θ and $E_{(x,y)\in D}[l(x,y;\theta)]$ is the expected loss on full dataset D when trained using the original model θ. Furthermore, the subset can be defined as $s_p = \{(x_i, y_i) \in D \mid \text{score}(x_i, y_i; \theta_p) \geq \tau\}$ where $\text{score}(x_i, y_i; \theta_p)$ represents the score assigned to each sample selected using θ_p. The score function can be based on various strategies such as Least Confidence, Entropy, or Greedy k-centers. τ defines the threshold used in the score-based selection methods (Least Confidence or Entropy) to determine the inclusion of a sample in s_p.

The goal of the optimization problem is to select s_p such that when θ is trained on it, the performance is as close as possible to training θ on the full dataset D. The key insight is that the subset s_p selected using the pruned model θ_p is sufficiently representative and informative for training the original model θ. This is because θ_p maintains a structure that is essentially identical to θ, although with some nodes pruned. As a result, there is a strong correlation between θ and θ_p, ensuring that the selection made by θ_p effectively minimizes the loss when θ is trained on s_p. By leveraging this surrogate θ_p, which is both computationally efficient and structurally coherent with θ, we can select most representative data out of D to train θ.

3.1 Pruning at Initialization

Pruning at initialization [39] shows potential in training time reduction, and enhanced model generalization. In our methodology, we employ structured pruning due to its benefits such as maintaining the architectural coherence of the network, enabling more predictable resource savings, and often leading to better-compressed models in practice. Consider an untrained neural network, represented as θ. Let each layer ℓ of this network have feature maps or channels denoted by c^ℓ, with $\ell \in \{1, \ldots, L\}$. Channel pruning results in binary masks $m^\ell \in \{0,1\}^{d^\ell}$ for every layer, where d^ℓ represents the total number of channels in layer ℓ. The pruned subnetwork, θ_p, retains channels described by $c^\ell \odot m^\ell$, where $\odot$ symbolizes the element-wise product. The sparsity $p \in [0,1]$ of the subnetwork illustrates the proportion of channels that are pruned: $p = 1 - \sum_\ell m^\ell / \sum_\ell d^\ell$. To reduce the model complexity, we employ channel pruning procedure $prune(C, p)$. This prunes to a sparsity p via two primary functions: i) score(C): This operation assigns scores $z^\ell \in \mathbb{R}^{d^\ell}$ to every channel in the network contingent on their magnitude (using the L2 norm). The channels C are represented as $(c_1, \ldots, c_L)$. and ii) remove(Z, p): This process takes the magnitude scores $Z = (z_1, \ldots, z_L)$ and translates them into masks m^ℓ such that the cumulative sparsity of the network, in terms of channels, is p. We employ a one-shot channel pruning that scores all the channels simultaneously based on their magnitude and prunes the network from 0% sparsity to $p\%$ sparsity in one cohesive step. Although previous works suggest re-initializing the network to ensure proper variance [40]. However, since the performance increment is marginal, we retain the weights of the pruned network before training.

3.2 Data Selection via Pruned Model

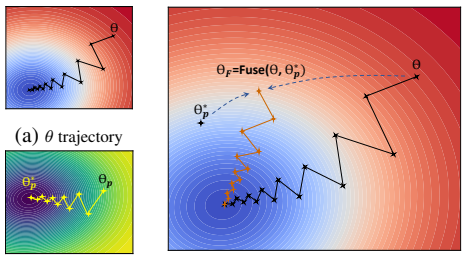
We begin by randomly selecting a small subset of data samples, denoted as s^0, from the unlabeled pool $U = \{x_i\}_{i\in[n]}$ where $[n] = \{1, ..., n\}$. These samples are then annotated. The pruned model θ_p is trained on this labeled subset s^0, resulting in the trained pruned model θ_p^*. With θ_p^* as our tool, we venture into the larger unlabeled dataset U to identify samples that are prime candidates for annotation. Regardless of the scenario, our method employs three distinct criteria for data selection: Least Confidence (LC) [41], Entropy [42], and Greedy k-centers [6]. LC based selection gravitates towards samples where the pruned model exhibits the least confidence in its predictions. Thus, the uncertainty score for a given sample x_i is defined as $\text{score}(x_i; \theta_p)_{\text{LC}} = 1 - \max_{\hat{y}} P(\hat{y}|x_i; \theta_p^*)$. The entropy-based selection focuses on samples with high prediction entropy, computed as $\text{score}(x_i; \theta_p)_{\text{Entropy}} = -\sum_{\hat{y}} P(\hat{y}|x_i; \theta_p^*) \log P(\hat{y}|\mathbf{x}_i; \theta_p^*)$, highlighting uncertainty. Subsequently, we select the top-k samples exhibiting the highest uncertainty scores, proposing them as prime candidates for annotation. The Greedy k-centers aims to cherry-pick k centers from the dataset such that the maximum distance of any sample from its nearest center is minimized. The selection is mathematically represented as $x = \arg\max_{x\in U} \min_{c\in\text{centers}} d(x, c)$ where centers is the current set of chosen centers and $d(x, c)$ is the distance between point x and center c. While various metrics can be employed to compute this distance, we opt for the Euclidean distance since it is widely used in this context.

3.3 Training of Pruned Model

Once we have selected the samples from U, they are annotated to obtain their respective labels. These freshly labeled samples are assimilated into the labeled dataset L. At the start of each training cycle, a fresh θ_p is generated. Training from scratch in every iteration is vital to prevent the model from developing spurious correlations or overfitting to specific samples [15]. This fresh start ensures that the model learns genuine patterns in the updated labeled dataset without carrying over potential biases from previous iterations. The training process adheres to a typical deep learning paradigm. Given the dataset L with samples (x_i, y_i), the aim is to minimize the loss function: $\mathcal{L}(\theta_p, L) = \frac{1}{|L|}\sum_{i=1}^{|L|} \mathcal{L}_i(\theta_p, x_i, y_i)$, where $\mathcal{L}_i$ denotes the individual loss for the sample x_i. Training unfolds over multiple iterations (or epochs). In each iteration, the weights of θ_p are updated using backpropagation with an optimization algorithm like stochastic gradient descent (SGD). This process is inherently iterative as in AL. After each round of training, new samples are chosen, annotated, and the model is reinitialized and retrained from scratch. This cycle persists until certain stopping criteria, e.g. labeling budget or desired performance, are met. With the incorporation of new labeled samples at every stage, θ_p^* progressively refines its performance, becoming better suited for the subsequent data selection phase.

3.4 Fusion with the Original Model

After achieving the predetermined budget, the next phase is to integrate the insights from the trained pruned model θ_p^* into the untrained original model θ. This step is crucial, as it amalgamates the learned knowledge from θ_p^* with the expansive architecture of the original model, aiming to harness the best of both worlds.



(a) θ trajectory

(b) θ_p trajectory (c) θ_F with a refined trajectory due to fusion

Figure 2: **Evolution of training trajectories**. Pruning θ to θ_p tailors the loss landscape from 2a to 2b, allowing θ_p to converge on an optimal configuration, denoted as θ_p^*. This model, θ_p^*, is later fused with the original θ, which provides better initialization and offers superior trajectory for θ_F to follow, as depicted in 2c.

Rationale for Fusion. Traditional pruning and fine-tuning methods often involve training a large model, pruning it down, and then fine-tuning the smaller model. While this is effective, it does not fully exploit the potential benefits of the larger, untrained model. The primary reason is that the pruning process might discard useful structures and connections within the original model that were not yet leveraged during initial training. By fusing the trained pruned model with the untrained original model, we aim to create a model that combines the learned knowledge by θ_p^* with the broader, unexplored model θ.

Method	Params	CIFAR-10 Label Budget (b)					CIFAR-100 Label Budget (b)					Params	Tiny-ImageNet-200 Label Budget (b)				
	(Million)	**10%**	**20%**	**30%**	**40%**	**50%**	**10%**	**20%**	**30%**	**40%**	**50%**	(Million)	**10%**	**20%**	**30%**	**40%**	**50%**
Baseline (AL)	0.85	80.53	87.74	90.85	92.24	93.00	35.99	52.99	59.29	63.68	66.72	25.56	14.86	33.62	43.96	49.86	54.65
PruneFuse ($p = 0.5$)	0.21	**80.92**	**88.35**	**91.44**	**92.77**	**93.65**	**40.26**	**53.90**	**60.80**	**64.98**	**67.87**	6.10	**18.71**	**39.70**	**47.41**	**51.84**	**55.89**
PruneFuse ($p = 0.6$)	0.13	**80.58**	87.79	90.94	**92.58**	**93.08**	**37.82**	52.65	**60.08**	**63.7**	**66.89**	3.92	**19.25**	**38.84**	**47.02**	**52.09**	**55.29**
PruneFuse ($p = 0.7$)	0.07	80.19	**87.88**	90.70	**92.44**	**93.40**	**36.76**	52.15	**59.33**	63.65	**66.84**	2.23	**18.32**	**39.24**	**46.45**	**52.02**	**55.63**
PruneFuse ($p = 0.8$)	0.03	80.11	87.58	90.50	**92.42**	**93.32**	**36.49**	50.98	58.53	62.87	65.85	1.02	**18.34**	**37.86**	**47.15**	**51.77**	**55.18**

Table 1: **Performance Comparison** of Baseline and PruneFuse on CIFAR-10, CIFAR-100 and Tiny ImageNet-200. This table summarizes the test accuracy of final models (original in case of AL and Fused in PruneFuse) for various pruning ratios (p) and labeling budgets(b). Least Confidence is used as a metric for subset selection and different architectures (ResNet-56 for CIFAR-10 and CIFAR-100 while ResNet-50 for Tiny-ImageNet) are utilized.

The Fusion Process. Fusion is executed by transferring the weights from the trained pruned model's weight matrix θ_p^* to the corresponding locations within the weight matrix of the untrained original model θ. This results in a new, fused weight matrix: $\theta_F = Fuse(\theta, \theta_p^*)$. Let's represent a model θ as a sequence of layers, where each layer L consists of filters (for CNNs). We can denote the i^{th} filter of layer j in model θ as $F_{i,j}^{\theta}$. Given: θ is the original untrained model and θ_p^* is the trained pruned model. For a specific layer j, θ has a set of n filters $\{F_{1,j}^{\theta}, F_{2,j}^{\theta}, ...F_{n,j}^{\theta}\}$ and θ_p^* has a set of m filters $\{F_{1,j}^{\theta_p^*}, F_{2,j}^{\theta_p^*}, ...F_{m,j}^{\theta_p^*}\}$ where $m \leq n$ due to pruning. The fusion process for layer j can be mathematically represented as:

$$F_{i,j}^{\theta_F} = \begin{cases} F_{i,j}^{\theta_p^*} & \text{if } F_{i,j}^{\theta_p^*} \text{ exists} \\ F_{i,j}^{\theta} & \text{otherwise} \end{cases}$$

Where $F_{i,j}^{\theta_F}$ is the i^{th} filter of layer j in the fused model θ_F.

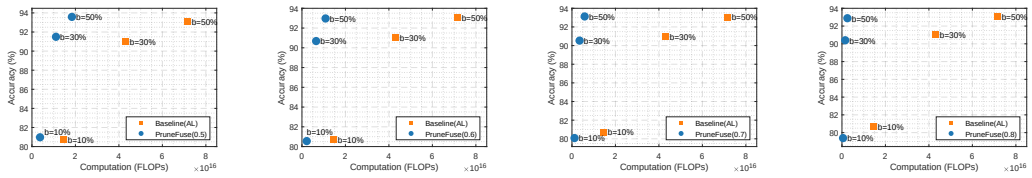
Advantages of Retaining Unaltered Weights: By copying weights from the trained pruned model θ_p^* into their corresponding locations within the untrained original model θ, and leaving the remaining weights of θ yet to be trained, we create a unique blend. The weights from θ_p^* encapsulate the knowledge acquired during training, providing a foundation. Meanwhile, the rest of the untrained weights in θ still have their initial values, offering an element of randomness. This duality fosters a richer exploration of the loss landscape during subsequent training. Fig. 2 illustrates the transformation in training trajectories resulting from the fusion process. The trained weights of θ_p^* provides a better initialization, while the unaltered weights serve as gateways to unexplored regions in the loss landscape. This strategic combination in the fused model θ_F enables the discovery of potentially superior solutions that neither the pruned nor the original model might have discovered on their own.

3.5 Refinement via Knowledge Distillation

After the fusion process, our resultant model, θ_F, embodies a synthesis of insights from both the trained pruned model θ_p^* and the original model θ. Although PruneFuse outperforms baseline AL (results are provided in Appendix), we further optimize and enhance θ_F using Knowledge Distillation (KD). KD enables θ_F to learn from θ_p^* (the teacher model), enriching its training. During the fine-tuning phase, we use two losses: i) Cross-Entropy Loss, which quantifies the divergence between the predictions of θ_F and the actual labels in dataset L, and ii) Distillation Loss, which measures the difference in the softened logits of θ_F and θ_p^*. These softened logits are derived by tempering logits of θ_p^*, which in our case is the teacher model, with a temperature parameter before applying the softmax function. The composite loss is formulated as a weighted average of both losses. The iterative enhancement of θ_F is governed by: $\theta_F^{(t+1)} = \theta_F^{(t)} - \alpha \nabla_{\theta_F^{(t)}} \Big(\lambda \mathcal{L}_{\text{Cross Entropy}}(\theta_F^{(t)}, L) + (1-\lambda) \mathcal{L}_{\text{Distillation}}(\theta_F^{(t)}, \theta_p^*) \Big)$. Here α represents the learning rate, while λ functions as a coefficient to balance the contributions of the two losses. By incorporating KD in the fine-tuning phase, we aim to ensure that the fused model θ_F not only retains the trained weights of pruned model but also reinforce this knowledge iteratively, optimizing the performance of θ_F in subsequent tasks.

4 Experiments

Experimental Setup. The effectiveness of our approach is assessed on three image classification datasets; CIFAR-10 [43], CIFAR-100 [43], and TinyImageNet-200 [44]. We used ResNet-50,



(a) (b) (c) (d)

Figure 3: **Computation Comparison of PruneFuse and Baseline (Active Learning):** This figure illustrates the total number of FLOPs utilized by PruneFuse, compared to the baseline Active Learning method, for selecting subsets with specific labeling budgets $b = 10\%, 30\%, 50\%$. The experiments are conducted on the CIFAR-10 dataset using the ResNet-56 architecture. Subfigures (a), (b), (c), and (d) correspond to different pruning ratios (0.5, 0.6, 0.7, and 0.8, respectively).

Method	Model	Architecture	No. of Parameters (Million)	Label Budget (b)				
				10%	**20%**	**30%**	**40%**	**50%**
SVP	Data Selector	ResNet-20	0.26	81.07	86.51	89.77	91.08	91.61
	Target	ResNet-56	0.85	80.76	87.31	90.77	92.59	92.95
PruneFuse	Data Selector	ResNet-56 ($p = 0.5$))	**0.21**	78.62	84.92	88.17	89.93	90.31
	Target	ResNet-56	0.85	**82.68**	**88.97**	**91.63**	**93.24**	**93.69**

Table 2: Performance Comparison of SVP and PruneFuse across various labeling budgets b for efficient training of Target Model (ResNet-56).

ResNet-56 and ResNet-164 architecture in our experiments. We pruned these architectures using the Torch-Prunnig library [45] for different pruning ratios $p = 0.5, 0.6, 0.7$, and 0.8 to get the pruned architectures. We trained the model for 181 epochs using the mini-batch of 128 for CIFAR-10 and CIFAR-100 and 100 epochs using the mini-batch of 256 for TinyImageNet-200. For all the experiments SGD is used as an optimizer. We took AL as a baseline for the proposed technique and initially, we started by randomly selecting 2% of the data. For the first round, we added 8% from the unlabeled set, then 10% in each subsequent round, until reaching the label budget, b. After each round, we retrained the models from scratch, as described in the methodology. All experiments are carried out independently 3 times and then the average is reported.

4.1 Results and Discussions

Main Experiments. Table 1 summarizes the generalization performance of baseline and different variants of PruneFuse on different datasets (detailed results on different architectures and data selection metrics are provided in Appendix). All variants of PruneFuse achieve higher accuracy compared to the baseline, demonstrating the effectiveness of superior data selection performance and fusion. Fig. 3 (a), (b), (c), and (d) illustrates the computational complexity of the baseline and PruneFuse variants in terms of Floating Point Operations (FLOPs) for different labeling budgets. The FLOPs are computed for the whole training duration of the pruned network and the selection process. Different variants of PruneFuse $p = 0.5, 0.6, 0.7$, and 0.8 provide the flexibility that the user can choose the variant of PruneFuse depending on their computation resources e.g. PruneFuse ($p = 0.8$) requires very low computation resources compared to others while achieving good accuracy performance.

Comparison with Selection-via-Proxy. Table 2 delineates a comparison of PruneFuse and the SVP [15], performance metrics show that PruneFuse consistently outperforms SVP across all labeling budgets for the efficient training of a Target Model (ResNet-56). SVP employs a ResNet-20 as its data selector, with a model size of 0.26 M. In contrast, PruneFuse uses a 50% pruned ResNet-56, reducing its data selector size to 0.21 M. Notably, while the data selector of PruneFuse achieves a lower accuracy of 90.31% at $b = 50\%$ compared to SVP's 91.61%, the target model utilizing PruneFuse-selected data attains a superior accuracy of 93.69%, relative to 92.95% for the SVP-selected data. This disparity underscores the distinct operational focus of the data selectors: PruneFuse's selector is optimized for enhancing the target model's performance, rather than its own.

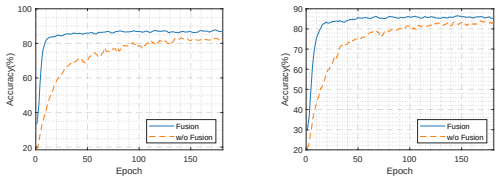
Ablation Studies. Table 3 demonstrates the effect of Knowledge Distillation (KD) on the PruneFuse technique relative to the baseline method across various data selection matrices and label budgets on CIFAR-100 datasets, using ResNet-56 architecture. The results indicate that PruneFuse consistently outperforms the baseline method, both with and without incorporating KD from a trained pruned

Method	Selection Metric	Label Budget (b)				
		10%	**20%**	**30%**	**40%**	**50%**
Baseline AL	Least Conf	35.99	52.99	59.29	63.68	66.72
	Entropy	37.57	52.64	58.87	63.97	66.78
	Random	37.06	51.62	58.77	62.05	64.63
	Greedy k	38.28	52.43	58.96	63.56	66.3
PruneFuse $p = 0.5$ (without KD)	Least Conf	**39.27**	**54.25**	**60.6**	**64.17**	**67.49**
	Entropy	**37.43**	**52.57**	**60.57**	**64.44**	**67.31**
	Random	**40.07**	**52.83**	**59.93**	**63.06**	**65.41**
	Greedy k	**39.25**	**52.43**	**59.94**	**63.94**	**66.56**
PruneFuse $p = 0.5$ (with KD)	Least Conf	**40.26**	**53.90**	**60.80**	**64.98**	**67.87**
	Entropy	**38.59**	**54.01**	**60.52**	**64.83**	**67.67**
	Random	**39.43**	**54.60**	**60.13**	**63.91**	**66.02**
	Greedy k	**39.83**	**54.35**	**60.40**	**64.22**	**66.89**

Table 3: Ablation Study of Knowledge Distillation on PruneFuse for CIFAR-100 datasets on Resnet-56

model. This superior performance is attributed to the innovative fusion strategy inherent to PruneFuse. The proposed approach gives the fused model an optimized starting point, enhancing its ability to learn more efficiently and generalize better. The impact of this strategy is evident across different label budgets and architectures, demonstrating its effectiveness and robustness.

Fig. 4 demonstrates the effect of fusion across various pruning ratios, the models trained with fusion in-place perform better than those trained without fusion, achieving higher accuracy levels at an accelerated pace. The rapid convergence is most notable in initial training phases, where fusion model benefits from the initialization provided by the integration of weights from a trained pruned model θ_p^* with an untrained model θ. The strategic retention of untrained weights introduces a beneficial stochastic component to the training process, enhancing the model's ability to explore new regions of the parameter space. This dual capability of exploiting prior knowledge and exploring new configurations enables the proposed technique to consistently outperform, making it particularly beneficial in scenarios with sparse label data.



(a) $p = 0.5, b = 30\%$ (b) $p = 0.6, b = 30\%$

Figure 4: **Impact of Model Fusion on PruneFuse Performance:** This figure compares the accuracy over epochs between fused and non-fused training approaches within the PruneFuse framework, both utilizing subset (with labeling budget b) selected by the pruned model. Experiments are conducted using the ResNet-56 on the CIFAR-10. Subfigures (a) and (b) correspond to pruning ratios $p = 0.5$ and 0.6, respectively.

5 Conclusion

We introduce PruneFuse, a novel approach combining pruning and network fusion to optimize data selection in deep learning. PruneFuse leverages a small pruned model for data selection, which then seamlessly fuses with the original model, providing fast and better generalization while significantly reducing computational costs. Extensive evaluations on CIFAR-10, CIFAR-100, and Tiny-ImageNet-200 show that PruneFuse outperforms existing baselines, establishing its efficiency and efficacy. PruneFuse offers a scalable, practical, and flexible solution to enhance the training efficiency of neural networks, particularly in resource-constrained settings.

6 Acknowledgments

This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIT) (No. RS-2024-00340966), and by Center for Applied Research in Artificial Intelligence (CARAI) grant funded by DAPA and ADD (UD230017TD).

References

[1] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. *Advances in neural information processing systems*, 28, 2015.

[2] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3431–3440, 2015.

[3] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[4] Chen Sun, Abhinav Shrivastava, Saurabh Singh, and Abhinav Gupta. Revisiting unreasonable effectiveness of data in deep learning era. In *Proceedings of the IEEE international conference on computer vision*, pages 843–852, 2017.

[5] Yarin Gal, Riashat Islam, and Zoubin Ghahramani. Deep bayesian active learning with image data. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 1183–1192. JMLR. org, 2017.

[6] Ozan Sener and Silvio Savarese. Active learning for convolutional neural networks: A core-set approach. *arXiv preprint arXiv:1708.00489*, 2017.

[7] Andreas Kirsch, Joost Van Amersfoort, and Yarin Gal. Batchbald: Efficient and diverse batch acquisition for deep bayesian active learning. *Advances in neural information processing systems*, 32, 2019.

[8] Yanyao Shen, Hyokun Yun, Zachary Lipton, Yakov Kronrod, and Animashree Anandkumar. Deep active learning for named entity recognition. In *Proceedings of the 2nd Workshop on Representation Learning for NLP*, pages 252–256, Vancouver, Canada, August 2017. Association for Computational Linguistics. doi: 10.18653/v1/W17-2630. URL `https://www.aclweb.org/anthology/W17-2630`.

[9] Ozan Sener and Silvio Savarese. Active learning for convolutional neural networks: A core-set approach. In *International Conference on Learning Representations*, 2018. URL `https://openreview.net/forum?id=H1aIuk-RW`.

[10] Alexander Freytag, Erik Rodner, and Joachim Denzler. Selecting influential examples: Active learning with expected model output changes. In *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part IV 13*, pages 562–577. Springer, 2014.

[11] Christoph Käding, Erik Rodner, Alexander Freytag, and Joachim Denzler. Active and continuous exploration with deep neural networks and expected model output changes. *arXiv preprint arXiv:1612.06129*, 2016.

[12] Krishnateja Killamsetty, Sivasubramanian Durga, Ganesh Ramakrishnan, Abir De, and Rishabh Iyer. Grad-match: Gradient matching based data subset selection for efficient deep model training. In *International Conference on Machine Learning*, pages 5464–5474. PMLR, 2021.

[13] Krishnateja Killamsetty, Durga Subramanian, Ganesh Ramakrishnan, and Rishabh Iyer. Glister: A generalization based data selection framework for efficient and robust learning. AAAI, 2021.

[14] Eeshaan Jain, Tushar Nandy, Gaurav Aggarwal, Ashish Tendulkar, Rishabh Iyer, and Abir De. Efficient data subset selection to generalize training across models: Transductive and inductive networks. *Advances in Neural Information Processing Systems*, 36, 2024.

[15] Cody Coleman, Christopher Yeh, Stephen Mussmann, Baharan Mirzasoleiman, Peter Bailis, Percy Liang, Jure Leskovec, and Matei Zaharia. Selection via proxy: Efficient data selection for deep learning. *arXiv preprint arXiv:1906.11829*, 2019.

[16] Barret Zoph and Quoc V Le. Neural architecture search with reinforcement learning. *arXiv preprint arXiv:1611.01578*, 2016.

[17] Alvin Wan, Xiaoliang Dai, Peizhao Zhang, Zijian He, Yuandong Tian, Saining Xie, Bichen Wu, Matthew Yu, Tao Xu, Kan Chen, et al. Fbnetv2: Differentiable neural architecture search for spatial and channel dimensions. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 12965–12974, 2020.

[18] Song Han, Huizi Mao, and William J Dally. Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding. *arXiv preprint arXiv:1510.00149*, 2015.

[19] Zhen Dong, Zhewei Yao, Daiyaan Arfeen, Amir Gholami, Michael W Mahoney, and Kurt Keutzer. Hawq-v2: Hessian aware trace-weighted quantization of neural networks. *Advances in neural information processing systems*, 33:18518–18529, 2020.

[20] Benoit Jacob, Skirmantas Kligys, Bo Chen, Menglong Zhu, Matthew Tang, Andrew Howard, Hartwig Adam, and Dmitry Kalenichenko. Quantization and training of neural networks for efficient integer-arithmetic-only inference. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2704–2713, 2018.

[21] Shuchang Zhou, Yuxin Wu, Zekun Ni, Xinyu Zhou, He Wen, and Yuheng Zou. Dorefa-net: Training low bitwidth convolutional neural networks with low bitwidth gradients. *arXiv preprint arXiv:1606.06160*, 2016.

[22] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.

[23] Hongxu Yin, Pavlo Molchanov, Jose M Alvarez, Zhizhong Li, Arun Mallya, Derek Hoiem, Niraj K Jha, and Jan Kautz. Dreaming to distill: Data-free knowledge transfer via deepinversion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8715–8724, 2020.

[24] Jongsoo Park, Sheng Li, Wei Wen, Ping Tak Peter Tang, Hai Li, Yiran Chen, and Pradeep Dubey. Faster cnns with direct sparse convolutions and guided pruning. *arXiv preprint arXiv:1608.01409*, 2016.

[25] Xin Dong, Shangyu Chen, and Sinno Pan. Learning to prune deep neural networks via layer-wise optimal brain surgeon. *Advances in neural information processing systems*, 30, 2017.

[26] Yiwen Guo, Anbang Yao, and Yurong Chen. Dynamic network surgery for efficient dnns. *Advances in neural information processing systems*, 29, 2016.

[27] Sejun Park, Jaeho Lee, Sangwoo Mo, and Jinwoo Shin. Lookahead: A far-sighted alternative of magnitude-based pruning. *arXiv preprint arXiv:2002.04809*, 2020.

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

[29] Yihui He, Xiangyu Zhang, and Jian Sun. Channel pruning for accelerating very deep neural networks. In *Proceedings of the IEEE international conference on computer vision*, pages 1389–1397, 2017.

[30] Zhonghui You, Kun Yan, Jinmian Ye, Meng Ma, and Ping Wang. Gate decorator: Global filter pruning method for accelerating deep convolutional neural networks. *Advances in neural information processing systems*, 32, 2019.

[31] Xiaohan Ding, Guiguang Ding, Yuchen Guo, and Jungong Han. Centripetal sgd for pruning very deep convolutional networks with complicated structure. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4943–4953, 2019.

[32] 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.

[33] Alex Renda, Jonathan Frankle, and Michael Carbin. Comparing rewinding and fine-tuning in neural network pruning. *arXiv preprint arXiv:2003.02389*, 2020.

[34] Michael Zhu and Suyog Gupta. To prune, or not to prune: exploring the efficacy of pruning for model compression. *arXiv preprint arXiv:1710.01878*, 2017.

[35] Trevor Gale, Erich Elsen, and Sara Hooker. The state of sparsity in deep neural networks. *arXiv preprint arXiv:1902.09574*, 2019.

[36] Namhoon Lee, Thalaiyasingam Ajanthan, and Philip HS Torr. Snip: Single-shot network pruning based on connection sensitivity. *arXiv preprint arXiv:1810.02340*, 2018.

[37] Jonathan Frankle, Gintare Karolina Dziugaite, Daniel M Roy, and Michael Carbin. Pruning neural networks at initialization: Why are we missing the mark? *arXiv preprint arXiv:2009.08576*, 2020.

[38] Hidenori Tanaka, Daniel Kunin, Daniel L Yamins, and Surya Ganguli. Pruning neural networks without any data by iteratively conserving synaptic flow. *Advances in neural information processing systems*, 33:6377–6389, 2020.

[39] Yulong Wang, Xiaolu Zhang, Lingxi Xie, Jun Zhou, Hang Su, Bo Zhang, and Xiaolin Hu. Pruning from scratch. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 12273–12280, 2020.

[40] Joost van Amersfoort, Milad Alizadeh, Sebastian Farquhar, Nicholas Lane, and Yarin Gal. Single shot structured pruning before training. *arXiv preprint arXiv:2007.00389*, 2020.

[41] Burr Settles. Active learning. *Synthesis Lectures on Artificial Intelligence and Machine Learning*, 6(1):1–114, 2012.

[42] Claude Elwood Shannon. A mathematical theory of communication. *The Bell system technical journal*, 27(3):379–423, 1948.

[43] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

[44] Ya Le and Xuan Yang. Tiny imagenet visual recognition challenge. *CS 231N*, 7(7):3, 2015.

[45] 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.

A Appendix

A.1 Performance Comparison with different Datasets, Selection Metrics, and Architectures

To comprehensively evaluate the effectiveness of PruneFuse, we conducted additional experiments comparing its performance with baseline utilizing other data selection metrics such as Least Confidence, Entropy, and Greedy k-centers. Results are shown in Tables 4 and 5 for various architectures and labeling budgets. In all cases, our results demonstrate that PruneFuse mostly outperforms the baseline using these traditional metrics across various datasets and model architectures, highlighting the robustness of PruneFuse in selecting the most informative samples efficiently.

Method	Selection Metric	Label Budget (b)				
		10%	**20%**	**30%**	**40%**	**50%**
Baseline AL	Least Conf	80.53	87.74	90.85	92.24	93.00
	Entropy	80.14	87.63	90.80	92.51	92.98
	Random	78.55	85.26	88.13	89.81	91.20
	Greedy k	79.63	86.46	90.09	91.9	92.80
PruneFuse $p = 0.5$	Least Conf	80.92	88.35	91.44	92.77	93.65
	Entropy	81.08	88.74	91.33	92.78	93.48
	Random	80.43	86.28	88.75	90.36	91.42
	Greedy k	79.85	86.96	90.20	91.82	92.89
PruneFuse $p = 0.6$	Least Conf	80.58	87.79	90.94	92.58	93.08
	Entropy	80.96	87.89	91.22	92.56	93.19
	Random	79.19	85.65	88.27	90.13	91.01
	Greedy k	79.54	86.16	89.5	91.35	92.39
PruneFuse $p = 0.7$	Least Conf	80.19	87.88	90.70	92.44	93.40
	Entropy	79.73	87.85	90.94	92.41	93.39
	Random	78.76	85.5	88.31	89.94	90.87
	Greedy k	78.93	85.85	88.96	90.93	92.23
PruneFuse $p = 0.8$	Least Conf	80.11	87.58	90.50	92.42	93.32
	Entropy	79.83	87.5	90.52	92.24	93.15
	Random	78.77	85.64	88.45	89.88	91.21
	Greedy k	78.23	85.59	88.60	90.11	91.31

(a) CIFAR-10.

Method	Selection Metric	Label Budget (b)				
		10%	**20%**	**30%**	**40%**	**50%**
Baseline AL	Least Conf	35.99	52.99	59.29	63.68	66.72
	Entropy	37.57	52.64	58.87	63.97	66.78
	Random	37.06	51.62	58.77	62.05	64.63
	Greedy k	38.28	52.43	58.96	63.56	66.30
PruneFuse $p = 0.5$	Least Conf	40.26	53.90	60.80	64.98	67.87
	Entropy	38.59	54.01	60.52	64.83	67.67
	Random	39.43	54.60	60.13	63.91	66.02
	Greedy k	39.83	54.35	60.40	64.22	66.89
PruneFuse $p = 0.6$	Least Conf	37.82	52.65	60.08	63.7	66.89
	Entropy	38.01	51.91	59.18	63.53	66.88
	Random	38.27	52.85	58.68	62.28	65.2
	Greedy k	38.44	52.85	59.36	63.36	66.12
PruneFuse $p = 0.7$	Least Conf	36.76	52.15	59.33	63.65	66.84
	Entropy	36.95	50.64	58.45	62.27	65.88
	Random	37.3	51.66	58.79	62.67	65.08
	Greedy k	38.88	52.02	58.66	61.39	65.28
PruneFuse $p = 0.8$	Least Conf	36.49	50.98	58.53	62.87	65.85
	Entropy	36.02	51.23	57.44	62.65	65.76
	Random	37.37	52.06	58.19	62.19	64.77
	Greedy k	37.04	49.84	56.13	60.24	62.92

(b) CIFAR-100.

Table 4: **Performance Comparison** of Baseline and PruneFuse on CIFAR-10 and CIFAR-100 with ResNet-56 architecture. This table summarizes the test accuracy of final models (original in case of AL and Fused in PruneFuse) for various pruning ratios (p), labeling budgets (b), and data selection metrics.

Method	Selection Metric	Label Budget (b)				
		10%	**20%**	**30%**	**40%**	**50%**
Baseline AL	Least Conf.	81.15	89.4	92.72	94.09	94.63
	Entropy	80.99	89.54	92.45	94.06	94.49
	Random	80.27	87.00	89.94	91.57	92.78
	Greedy k	80.02	88.33	91.76	93.39	94.40
PruneFuse $p = 0.5$	Least Conf.	83.03	90.30	93.00	94.41	94.63
	Entropy	82.64	89.88	93.08	94.32	94.90
	Random	81.52	87.84	90.14	91.94	92.81
	Greedy k	81.70	88.75	91.92	93.64	94.22
PruneFuse $p = 0.6$	Least Conf.	82.86	90.22	93.05	94.27	94.66
	Entropy	82.23	90.18	92.91	94.28	94.66
	Random	81.14	87.51	90.05	91.82	92.43
	Greedy k	81.11	88.41	91.66	92.94	94.17
PruneFuse $p = 0.7$	Least Conf.	82.76	89.89	92.83	94.10	94.69
	Entropy	82.59	89.81	92.77	94.20	94.74
	Random	80.88	87.54	90.09	91.57	92.64
	Greedy k	81.68	88.36	91.64	93.02	93.97
PruneFuse $p = 0.8$	Least Conf.	82.66	89.78	92.64	94.08	94.69
	Entropy	82.01	89.77	92.65	94.02	94.60
	Random	80.73	87.43	90.08	91.40	92.53
	Greedy k	79.66	87.56	90.79	92.30	93.17

(a) CIFAR-10.

Method	Selection Metric	Label Budget (b)				
		10%	**20%**	**30%**	**40%**	**50%**
Baseline AL	Least Conf	38.41	51.39	65.53	70.07	73.05
	Entropy	36.65	57.58	64.98	69.99	72.90
	Random	39.31	57.53	63.84	67.75	70.79
	Greedy k	39.76	57.40	65.20	69.25	72.91
PruneFuse $p = 0.5$	Least Conf	42.88	59.31	66.95	71.45	74.32
	Entropy	42.99	59.32	66.83	71.18	74.43
	Random	43.72	58.58	64.93	68.75	71.63
	Greedy k	43.61	58.38	66.04	69.83	73.10
PruneFuse $p = 0.6$	Least Conf	41.86	58.97	66.61	70.59	73.6
	Entropy	42.43	58.74	65.97	70.90	73.70
	Random	42.53	58.33	65.00	68.55	71.46
	Greedy k	42.71	58.41	65.43	69.57	72.49
PruneFuse $p = 0.7$	Least Conf	42.00	57.08	66.41	70.68	73.63
	Entropy	41.01	57.45	65.99	70.07	73.45
	Random	42.76	57.31	64.12	68.07	70.88
	Greedy k	42.42	57.58	65.18	68.55	71.89
PruneFuse $p = 0.8$	Least Conf	41.19	57.98	65.22	70.38	73.17
	Entropy	39.78	57.3	65.19	69.40	72.82
	Random	42.08	57.23	64.05	67.85	70.62
	Greedy k	42.20	57.42	64.53	68.01	71.29

(b) CIFAR-100.

Table 5: **Performance Comparison** of Baseline and PruneFuse on CIFAR-10 and CIFAR-100 with ResNet-164 architecture. This table summarizes the test accuracy of final models (original in case of AL and Fused in PruneFuse) for various pruning ratios (p), labeling budgets (b), and data selection metrics.

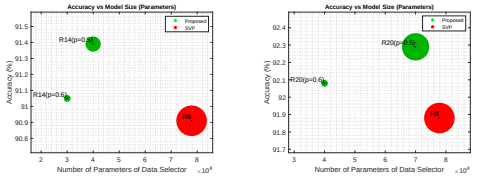
A.2 Comparison with SVP

Table 6 demonstrates the performance comparison of PruneFuse and SVP for small model architecture ResNet-20 on CIFAR-10. SVP achieves 91.88% performance accuracy by utilizing the data selector having 0.074 M parameters whereas PruneFuse outperforms SVP by achieving 92.29% accuracy with a data selector of 0.066 M parameters.

Fig. 5(a) and (b) show that target models when trained with the data selectors of the PruneFuse achieve significantly higher accuracy while using significantly less number of parameters compared to SVP. These results indicate that the PruneFuse does not require an additional architecture for designing the data selector; it solely needs the target model. In contrast, SVP necessitates both the target model (ResNet-14) and a smaller model (ResNet-8) that functions as a data selector.

Techniques	Model	Architecture	No. of Parameters (Million)	Label Budget (b)				
				10%	20%	30%	40%	50%
SVP	Data Selector	ResNet-8	0.074	77.85	83.35	85.43	86.83	86.90
	Target	ResNet-20	0.26	80.18	86.34	89.22	90.75	91.88
PruneFuse	Data Selector	ResNet-20 ($p = 0.5$)	**0.066**	76.58	83.41	85.83	87.07	88.06
	Target	ResNet-20	0.26	**80.25**	**87.57**	**90.20**	**91.70**	**92.29**

Table 6: Comparison of SVP and PruneFuse on Small Models.

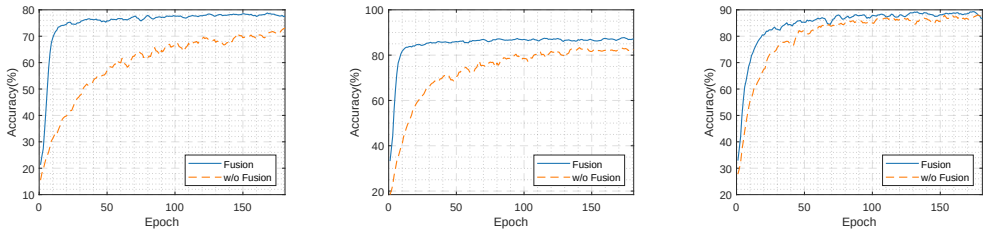


(a) Target Model = ResNet-14 (b) Target Model = ResNet-20

Figure 5: **Comparison of PruneFuse with SVP.** Scatter plot shows final accuracy on target model against the model size for different ResNet models on CIFAR-10 dataset with labeling budget $b = 50\%$. (a) shows for the target network ResNet-14, ResNet-14 (with $p = 0.5$ and $p = 0.6$) and ResNet-8 models are used as data selectors for PruneFuse and SVP, respectively. While in (b), PruneFuse utilizes ResNet20 (i.e. $p = 0.5$ and $p = 0.6$) and SVP utilizes ResNet-8 models for data selection when the target model is ResNet-20.

A.3 Ablation Study of Fusion

The fusion process is a critical component of the PruneFuse methodology, designed to integrate the knowledge gained by the pruned model into the original network. Our experiments reveal that models trained with the fusion process exhibit significantly better performance and faster convergence compared to those trained without fusion. By initializing the original model with the weights from the trained pruned model, the fused model benefits from an optimized starting point, which enhances its learning efficiency and generalization capability. Fig. 6 illustrates the training trajectories and accuracy improvements when fusion takes places, demonstrating the tangible benefits of this initialization. These results underscore the importance of the fusion step in maximizing the overall performance of the PruneFuse framework.



(a) $p = 0.5, b = 10\%$ (b) $p = 0.5, b = 30\%$ (c) $p = 0.5, b = 50\%$

Figure 6: **Ablation Study of Fusion on PruneFuse** ($p = 0.5$). Experiments are performed on ResNet-56 architecture with CIFAR-10.

A.4 Ablation Study of Knowledge Distillation in PruneFuse

Table 7 demonstrates the effect of Knowledge Distillation on the PruneFuse technique relative to the baseline Active Learning (AL) method across various experimental configurations and label budgets on CIFAR-10 and CIFAR-100 datasets, using ResNet-56 architecture. The results indicate that PruneFuse consistently outperforms the baseline method, both with and without incorporating Knowledge Distillation (KD) from a trained pruned model. This superior performance is attributed to the innovative fusion strategy inherent to PruneFuse, where the original model is initialized using weights from a previously trained pruned model. The proposed approach gives the fused model an optimized starting point, enhancing its ability to learn more efficiently and generalize better. The impact of this strategy is evident across different label budgets and architectures, demonstrating its effectiveness and robustness.

Method	Selection Metric	Label Budget (b)				
		10%	**20%**	**30%**	**40%**	**50%**
Baseline *AL*	Least Conf	80.53	87.74	90.85	92.24	93.00
	Entropy	80.14	87.63	90.80	92.51	92.98
	Random	78.55	85.26	88.13	89.81	91.20
	Greedy k	79.63	86.46	90.09	91.90	92.80
PruneFuse $p = 0.5$ (without KD)	Least Conf	**81.08**	**88.71**	**91.24**	**92.68**	**93.46**
	Entropy	**80.80**	**88.08**	**90.98**	**92.74**	**93.43**
	Random	**80.11**	**85.78**	**88.81**	**90.20**	91.10
	Greedy k	**80.07**	**86.70**	89.93	91.72	92.67
PruneFuse $p = 0.5$ (with KD)	Least Conf	**80.92**	**88.35**	**91.44**	**92.77**	**93.65**
	Entropy	**81.08**	**88.74**	**91.33**	**92.78**	**93.48**
	Random	**80.43**	**86.28**	**88.75**	**90.36**	**91.42**
	Greedy k	**79.85**	**86.96**	**90.20**	91.82	**92.89**

(a) CIFAR-10.

Method	Selection Metric	Label Budget (b)				
		10%	**20%**	**30%**	**40%**	**50%**
Baseline *AL*	Least Conf	35.99	52.99	59.29	63.68	66.72
	Entropy	37.57	52.64	58.87	63.97	66.78
	Random	37.06	51.62	58.77	62.05	64.63
	Greedy k	38.28	52.43	58.96	63.56	66.3
PruneFuse $p = 0.5$ (without KD)	Least Conf	**39.27**	**54.25**	**60.6**	**64.17**	**67.49**
	Entropy	**37.43**	**52.57**	**60.57**	**64.44**	**67.31**
	Random	**40.07**	**52.83**	**59.93**	**63.06**	**65.41**
	Greedy k	**39.25**	**52.43**	**59.94**	**63.94**	**66.56**
PruneFuse $p = 0.5$ (with KD)	Least Conf	**40.26**	**53.90**	**60.80**	**64.98**	**67.87**
	Entropy	**38.59**	**54.01**	**60.52**	**64.83**	**67.67**
	Random	**39.43**	**54.60**	**60.13**	**63.91**	**66.02**
	Greedy k	**39.83**	**54.35**	**60.40**	**64.22**	**66.89**

(b) CIFAR-100.

Table 7: Ablation Study of Knowledge Distillation on PruneFuse presented in (a), and (b) for different datasets on Resnet-56

A.5 Ablation of Different Selection Metrics

The impact of different selection metrics is presented in Table 8 demonstrating clear differences in performance across both the Baseline and PruneFuse methods on CIFAR-100 using ResNet-164 architecture. It is evident that across both the baseline and PruneFuse methods, the Least Confidence metric surfaces as particularly effective in optimizing label utilization and model performance. The results further reinforce that regardless of the label budget, from 10% to 50%, PruneFuse demonstrates a consistent superiority in performance with different data selection metrics (Least Confidence, Entropy, Random, and Greedy k-centres) compared to Baseline.

Method	Selection Metric	Label Budget (b)				
		10%	**20%**	**30%**	**40%**	**50%**
Baseline *AL*	Least Conf	38.41	51.39	65.53	70.07	73.05
	Entropy	36.65	57.58	64.98	69.99	72.90
	Random	39.31	57.53	63.84	67.75	70.79
	Greedy k	39.76	57.40	65.20	69.25	72.91
PruneFuse $p = 0.5$	Least Conf	**42.88**	**59.31**	**66.95**	**71.45**	**74.32**
	Entropy	**42.99**	**59.32**	**66.83**	**71.18**	**74.43**
	Random	**43.72**	**58.58**	**64.93**	**68.75**	**71.63**
	Greedy k	**43.61**	**58.38**	**66.04**	**69.83**	**73.10**

Table 8: Effect of Different Data Selection Metrics on CIFAR-100 using ResNet-164 architecture.

B Algorithmic Details

In this section, we provide a detailed explanation of the PruneFuse algorithm given in Algorithm 1. The PruneFuse methodology begins with structured pruning an untrained neural network, θ, to create a smaller model, θ_p. This pruning step reduces complexity while retaining the network's essential structure, allowing θ_p to efficiently select informative samples from the unlabeled dataset, U. The algorithm proceeds as follows. First, the original model θ is randomly initialized and pruned to obtain θ_p. The pruned model θ_p is then trained on an initial labeled dataset s^0 to produce θ_p^*. This training equips θ_p with preliminary knowledge for data selection. The labeled dataset L is initially set to s^0. A data selection loop runs until the labeled dataset L reaches the maximum budget b. In each iteration, θ_p is retrained on L to keep the model updated with new samples. Uncertainty scores for all samples in U are computed using the trained θ_p^* on the available labelel subset, and the top-k samples with the highest scores are selected as D_k. These samples are labeled and added to L. Once the budget b is met, the final trained pruned model θ_p^* is fused with the original model θ to create the fused model θ_F. This fusion transfers the weights from θ_p^* to θ, ensuring the pruned model's knowledge is retained. Finally, θ_F is trained on L using knowledge distillation from θ_p^*, refining the model's performance by leveraging the pruned model's learned insights. In summary, PruneFuse strategically adapts pruning in data selection problem and to enhance both data selection efficiency and model performance.

Algorithm 1 Efficient Data Selection using Pruned Networks

Input: Unlabeled dataset U, Initial labeled dataset s^0, labeled dataset L, original model θ, prune model θ_p, fuse model θ_F, maximum budget b, pruning ratio p.

Randomly initialize θ
$\theta_p \leftarrow$ Prune(θ, p) // Structure pruning
$\theta_p^* \leftarrow$ Train θ_p on s^0
$L \leftarrow s^0$
while $|L| \leq b$ **do**
    Compute score$(\mathbf{x}; \theta_p^*)$ for all $x \in U$ //Compute uncertainty scores for samples in U using θ_p^*
    $D_k = top_k[D_j \in U]_{j \in [k]}$ //Select top-k samples with highest uncertainty scores
    Query labels y_k for selected samples D_k
    Add (D_k, y_k) to L
    Train θ_p^* on L
$\theta_F \leftarrow Fuse(\theta, \theta_p^*)$
$\theta_F^* \leftarrow$ Train θ_F on L
return L, θ_F^*

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: the abstract and introduction accurately reflect the contributions that we developed PruneFuse for efficient data selection.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [No]

   Justification: We have not explicitly mentioned the limitations in a seperate section as we have strived to present a balanced and accurate view of our technique and contributions.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory Assumptions and Proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [NA]

Justification: This paper does not include theoretical analysis.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental Result Reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: the experimental section of the main paper contains all the implementation details needed to reproduce the main results.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [No]

Justification: We provide detailed description of implementation details and believe that it is easy to implement. However, we recognize the value of open-source practices and plan to publish the code on GitHub following publication.

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental Setting/Details**

Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

Answer: [Yes]

Justification: the training and test details are discussed in detail in the experimental section in the main text.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
- The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment Statistical Significance**

Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [Yes]

Justification: all experiments are performed with random seed values for three trials and the average is reported.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.

- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments Compute Resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: We provide detailed computation costs associated with data selection process in main text as well as in supplementary materials.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code Of Ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: our research does follow the NeurIPS code of ethics.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader Impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [NA]

    Justification: we discuss in detail about the need of efficient data selection strategy to reduce the overall computation costs for training deep neural networks. Our work reduces the overall computation costs significantly for general data selection frameworks which achieving good performance.

Guidelines:

- The answer NA means that there is no societal impact of the work performed.
- If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification: we pose no potential for misuse in our technique.

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: we accurately cite the related works and resources that were utilized in our paper.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.

- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New Assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [NA]

    Justification: we do not release new assets.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and Research with Human Subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: we do not involve any human subject in our technique/experiments.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional Review Board (IRB) Approvals or Equivalent for Research with Human Subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: the paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.

- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.