Bi-level Optimization for Model Pruning in Decentralized Environments

Heng Liang
Department of Computer and Information Science, Temple University
Philadelphia, US
tuv05559@temle.edu

Qi Chen
Department of Computer and Information Science, Temple University
Philadelphia, US
qi.chen0004@temle.edu

Abstract

Modern neural networks continue to grow in scale, creating substantial computational and communication challenges for decentralized learning systems characterized by heterogeneous clients and non-IID data distributions. Although pruning is a widely-used technique to reduce model size and communication overhead, most existing pruning approaches rely on heuristic rules and decouple pruning from model training, leading to suboptimal sparse structures and instability in federated or decentralized environments. To address these limitations, we propose a bi-level optimization framework that jointly learns the pruning mask and model parameters. In our method, the upper-level problem optimizes the pruning mask to improve global performance, while the lower-level problem trains model parameters under a fixed mask. This formulation enables a principled coupling between structural decisions and parameter updates, overcoming the limitations of traditional heuristic pruning.

We evaluate the proposed method on decentralized training tasks involving both RNN and linear SVM models across multiple datasets. Experimental results show that our approach consistently achieves faster convergence, higher AUC, and lower training loss compared with baseline pruning strategies. The improvements are particularly significant in heterogeneous or non-IID data settings, demonstrating the advantages of bi-level optimization for structure-aware model compression. Preliminary experiments on a CNN (ResNet-18) with CIFAR-10 indicate that larger models introduce additional optimization challenges, motivating further refinement of our algorithm. Overall, our findings confirm that bi-level pruning offers an effective, optimization-driven alternative to heuristic pruning in decentralized learning systems.

Keywords: Model Pruning, Bi-level Optimization, Decentralized, Distributed Training Communication Efficiency

1 Introduction

The rapid growth of modern neural networks has significantly improved performance across a wide range of tasks, but it also introduces substantial challenges in computation, storage, and communication [1, 5]. These challenges are particularly pronounced in decentralized and distributed learning settings, where multiple clients collaboratively train a model without sharing raw data [7]. In such environments, limited bandwidth, heterogeneous hardware, and non-IID data distributions make efficient model training and deployment increasingly difficult.

Model pruning is a widely adopted technique to address these issues by removing unimportant model parameters, thereby reducing model size and communication overhead [6]. However, most existing pruning methods are heuristic in nature, such as magnitude-based pruning, and are typically applied either after training or independently from the optimization process [4]. While effective in centralized settings, these approaches often suffer from performance degradation and instability when applied to decentralized learning scenarios. In particular, heuristic pruning does not explicitly account for client heterogeneity or non-IID data, which can amplify accuracy loss and slow convergence.

Recent studies suggest that the limitations of heuristic pruning stem from its decoupled design: the pruning decision and the model training process are treated as separate stages rather than being jointly optimized [1]. This observation motivates the exploration of optimization-driven pruning strategies. Bi-level optimization provides a principled framework to address this challenge by naturally modeling hierarchical decision-making problems [2]. In the context of model pruning, the upper-level problem determines the model structure (i.e., which parameters to prune), while the lower-level problem optimizes the remaining model parameters under a fixed structure.

In this project, we adopt a bi-level optimization perspective to design a pruning framework tailored for decentralized learning. The pruning mask is optimized at the upper level to improve global performance, while model parameters are trained at the lower level given the current mask. This joint formulation enables pruning decisions to be directly guided by training dynamics, rather than by static heuristics. As a result, the learned sparse models are better aligned with the underlying data distribution and training objectives.

We evaluate the proposed approach on decentralized learning tasks involving both RNN and SVM models across multiple datasets. Experimental results demonstrate that the bi-level pruning method achieves faster convergence, improved predictive performance, and greater robustness to non-IID data compared with baseline pruning strategies. These findings highlight the effectiveness of bi-level optimization as a structured and principled alternative to heuristic pruning in decentralized learning systems.

The remainder of this report is organized as follows. Section 2 reviews related work on model pruning and bi-level optimization. Section 3 introduces preliminary background on model pruning and bi-level optimization. Section 4 presents the proposed bi-level pruning method in detail. Section 5 describes the experimental setup and reports the evaluation results. Finally, Section 6 concludes the report and discusses future directions.

2 Related Works

In this section, we review prior work related to model pruning and bi-level optimization, which form the theoretical and methodological foundations of our approach.

2.1 Model pruning

Model pruning has been extensively studied as an effective technique for reducing the size and computational cost of neural networks [1, 5]. Early work primarily focused on heuristic-based approaches, such as magnitude pruning, which removes parameters with small absolute weights [6]. These methods are simple and computationally efficient, and have been widely adopted in centralized training settings. Variants of magnitude-based pruning have also been explored in iterative frameworks, most notably Iterative Magnitude Pruning (IMP), which progressively removes parameters while retraining the model to recover accuracy [4].

Despite their success, heuristic pruning methods exhibit several limitations. First, pruning decisions are typically decoupled from the training process, meaning that the model structure is not optimized jointly with model parameters. Second, many pruning strategies are designed for centralized training and assume homogeneous data distributions. When applied to decentralized or federated learning settings with non-IID data, these methods often suffer from degraded performance, unstable convergence, and reduced generalization [7]. Recent studies have shown that such heuristic approaches may fail to consistently identify high-quality sparse subnetworks, especially under high sparsity constraints or heterogeneous data regimes.

To address these challenges, optimization-based pruning methods have been proposed. These approaches formulate pruning as a constrained optimization problem by introducing sparsity regularization or learnable masks [8]. While these methods provide a more principled foundation than heuristic pruning, they often struggle to match the performance of iterative pruning schemes and may introduce additional optimization complexity.

2.2 Bi-level Optimization

Bi-level optimization is a hierarchical optimization framework consisting of two nested problems, where the solution of the upper-level problem depends on the optimal solution of the lower-level problem [2]. This formulation naturally arises in problems that involve structure selection or hyperparameter optimization, such as meta-learning and neural architecture search [3].

In the context of model pruning, bi-level optimization offers a natural way to decouple structural decisions from parameter optimization. The upper-level problem can be used to optimize the pruning mask or model structure, while the lower-level problem focuses on training model parameters under a fixed structure. This separation enables a principled interaction between pruning and training, allowing pruning decisions to be guided by training dynamics rather than static heuristics.

Recent work has demonstrated that model pruning can be reformulated as a bi-level optimization problem with favorable properties, such as bi-linearity between pruning masks and model parameters [9]. By exploiting this structure, bi-level pruning methods can achieve computational efficiency comparable to first-order optimization while maintaining strong performance, making them well-suited for decentralized and non-IID settings.

3 Preliminary

This section introduces the background concepts required to understand our proposed approach. We first review the basic formulation of model pruning, followed by an overview of bi-level optimization and its relevance to pruning problems.

3.1 Model Pruning

Model pruning aims to reduce the size and computational cost of a neural network by removing unimportant parameters while preserving predictive performance. Given a model with parameters $\mathbf{w} \in \mathbb{R}^n$, pruning is commonly formulated by introducing a binary mask $\mathbf{m} \in \{0, 1\}^n$, where each element of $\mathbf{m}$ determines whether the corresponding parameter in $\mathbf{w}$ is retained. The pruned model can be expressed as

$$\mathbf{w}' = \mathbf{m} \odot \mathbf{w}, \tag{1}$$

where $\odot$ denotes element-wise multiplication.

Traditional pruning approaches typically rely on heuristic criteria, such as weight magnitude, gradient magnitude, or sensitivity measures, to determine which parameters should be removed. These methods are often applied either after model training or iteratively with retraining steps. While effective in centralized settings, heuristic pruning does not explicitly account for the interaction between model structure and training dynamics, which can lead to suboptimal sparse models, especially under high sparsity or non-IID data distributions.

From the perspective of decentralized learning, pruning plays an additional role by reducing communication overhead among distributed clients. Smaller models lead to lower transmission costs during parameter aggregation. However, improper pruning may amplify performance gaps across clients with heterogeneous data, motivating the need for more principled pruning strategies.

3.2 Bi-level Optimization

Bi-level optimization is a hierarchical optimization framework consisting of two nested optimization problems. The upper-level problem optimizes a set of high-level variables, while the lower-level problem optimizes another set of variables whose solution depends on the upper-level variables. A general bi-level optimization problem can be written as

$$\min_{\mathbf{x}} F(\mathbf{x}, \mathbf{y}^*(\mathbf{x})) \quad \text{s.t.} \quad \mathbf{y}^*(\mathbf{x}) = \arg\min_{\mathbf{y}} G(\mathbf{x}, \mathbf{y}), \tag{2}$$

where $\mathbf{x}$ denotes the upper-level variable and $\mathbf{y}$ denotes the lower-level variable.

Bi-level optimization naturally fits problems that involve structure selection or hyperparameter tuning. In the context of model pruning, the pruning mask can be treated as the upper-level variable, while the model parameters are optimized at the lower level under a fixed mask. This formulation explicitly captures the dependency between pruning decisions and model training.

A key challenge in bi-level optimization lies in efficiently computing gradients for the upper-level problem, as the lower-level solution implicitly depends on the upper-level variables. Recent studies have shown that under certain structural assumptions, such as bi-linearity between variables, bi-level optimization can be solved efficiently using first-order methods. This observation motivates the use of bi-level optimization as a practical and principled foundation for model pruning, particularly in decentralized learning settings.

The concepts introduced in this section lay the groundwork for the bi-level pruning framework presented in the next section.

4 Method

In this section, we present our bi-level optimization framework for model pruning in decentralized environments. The key idea is to integrate pruning directly into decentralized training, so that the pruning mask and model parameters are optimized jointly while respecting data locality and communication constraints.

4.1 Problem Setting

We consider a decentralized learning setup with K workers / clients connected by a communication graph. The k-th worker holds a local dataset $\mathcal{D}_k$ drawn from a possibly non-IID distribution and defines a local loss function $\mathcal{L}^{(k)}(\cdot; \mathcal{D}_k)$. The goal is to collaboratively train a compact model that achieves high accuracy while reducing both model size and communication cost.

Let $y \in \mathbb{R}^d$ denote the model parameters and let $x \in \mathbb{R}^d$ denote the pruning parameters (or mask), with the same dimension as y. The effective parameters used for prediction are given by the element-wise product

$$\theta(x, y) \; = \; x \odot y \in \mathbb{R}^d,$$

which gates individual weights and induces sparsity in the model. This formulation can be instantiated for different architectures:

- For SVMs, y collects the linear classifier weights, and x prunes feature dimensions.
- For RNNs, y stacks recurrent and output-layer weights, and x controls sparsity in temporal connections.
- For CNNs, y corresponds to convolutional and fully connected filters, and x performs unstructured pruning by masking individual weights within these filters.

4.2 Bi-level Pruning Formulation

Following the spirit of bi-level pruning in centralized settings [?], we formulate decentralized pruning as a bi-level optimization problem. The upper-level problem optimizes the pruning mask x by trading off global performance and model compactness, while the lower-level problem optimizes the model parameters y for a fixed mask.

Concretely, the *lower-level problem* learns the model parameters y on top of a given pruning mask x:

$$y^*(x) \; = \; \arg\min_{y} \frac{1}{K} \sum_{k=1}^{K} \mathcal{L}^{(k)}\big(x \odot y \,;\, \mathcal{D}_k\big), \tag{3}$$

where the loss is computed locally on each worker using its own data and the pruned parameters $x \odot y$.

Given the implicit solution $y^*(x)$, the *upper-level problem* adjusts the pruning mask by minimizing the global performance after local training, together with a regularizer that encourages sparsity and communication efficiency:

$$\min_{x} F(x) \; := \; \frac{1}{K} \sum_{k=1}^{K} \mathcal{L}^{(k)}\big(x \odot y^*(x) \,;\, \mathcal{D}_k\big) \; + \; \lambda R(x), \tag{4}$$

where $R(x)$ is a sparsity-promoting regularizer (e.g., ℓ_1 or structured group penalties) and $\lambda > 0$ controls the pruning strength. Intuitively, (4) searches for a pruning policy that maintains accuracy while removing as many redundant parameters as possible and indirectly reducing communication cost.

Together, (3) and (4) define a nonconvex bi-level optimization problem in the decentralized setting, which is significantly more challenging than its centralized counterpart due to data heterogeneity and limited communication.

4.3 Decentralized Bi-level Optimization Algorithm

Solving (4)–(3) exactly is infeasible in practice, so we adopt an approximate scheme that alternates between *inner* updates of y and *outer* updates of x.

Inner loop (lower level). For a fixed pruning mask x, each worker performs multiple local stochastic gradient steps to minimize its local objective with respect to y. Denoting by $y_t^{(k)}$ the local copy of the model parameters at iteration t on worker k, the update takes the form

$$y_{t+1}^{(k)} = \sum_{j \in \mathcal{N}_k} w_{kj}\, y_t^{(j)} - \eta_y\, g_t^{(k)},$$

where $\mathcal{N}_k$ denotes the neighbors of worker k in the communication graph, (w_{kj}) are mixing weights, η_y is the step size, and $g_t^{(k)}$ is a stochastic gradient of $\mathcal{L}^{(k)}(x \odot y; \mathcal{D}_k)$ evaluated at $y_t^{(k)}$. We employ variance-reduced gradient tracking to stabilize training under non-IID data and reduce the number of communication rounds.

Outer loop (upper level). After a certain number of inner-loop steps, the workers collaboratively update the pruning mask x. Given gradient estimators $\{\nabla_x \mathcal{L}^{(k)}(x \odot y_t^{(k)})\}_{k=1}^{K}$, we approximate the hypergradient of $F(x)$ and perform a projected gradient step:

$$x_{t+1} = \mathcal{P}_X\left(x_t - \eta_x \left(\frac{1}{K}\sum_{k=1}^{K} \nabla_x \mathcal{L}^{(k)}(x_t \odot y_t^{(k)}) + \lambda\, \nabla R(x_t)\right)\right),$$

where η_x is the outer-loop step size, $\mathcal{P}_X$ denotes projection onto a feasible set (e.g., box constraints or probability simplex for normalized masks), and the gradients are aggregated in a decentralized manner via the same communication graph.

4.4 Communication Efficiency

The proposed framework improves communication efficiency in two complementary ways:

- **Parameter sparsity.** As training progresses, the learned mask x drives many entries of $x \odot y$ towards zero, enabling sparse communication of model updates.
- **Decentralized updates.** Both the inner and outer loops rely only on neighbor-to-neighbor communication, avoiding any centralized parameter server and naturally fitting multi-agent or federated scenarios.

5 Experiments

In this section, we empirically evaluate the proposed decentralized bi-level pruning framework on three representative model families: SVMs, RNNs, and CNNs.

5.1 Datasets and Models

We consider the following models and datasets.

- **SVMs.** We use a simple two-layer linear classifier on several standard binary classification benchmarks, including a9a, covtype, and IMDB. Concretely, the model first applies a linear feature transformation with weight matrix $W_1 \in \mathbb{R}^{d \times H}$, followed by a linear classifier $W_2 \in \mathbb{R}^{H \times 2}$ for binary prediction. The trainable parameters y are obtained by concatenating the flattened weights of W_1 and W_2, while the pruning parameter x is an unstructured mask applied to W_1 to prune individual weights.
- **RNNs.** We adopt a recurrent neural network classifier on the Sent140 dataset to study how pruning interacts with temporal dependencies and convergence stability in sequence models. Each input tweet is represented as a sequence of 300-dimensional word embeddings, forming an input tensor of shape $[T, B, 300]$, where T is the sequence length and B is the batch size. The classifier is implemented as a two-layer Simple RNN with hidden size H (e.g., $H = 4096$), followed by a fully connected layer that maps the final hidden representation to two output logits.
- **CNNs.** We use a ResNet-18 architecture on CIFAR-10 as a representative convolutional neural network. In this setting, we apply *unstructured* pruning by masking individual weights in the convolutional and fully connected layers.

For completeness, we note the division of implementation work. Heng Liang is primarily responsible for experiments on Sent140, a9a, and CIFAR-10, while Qi Chen focuses on covtype and IMDB. Both authors jointly analyze the results and prepare the final report.

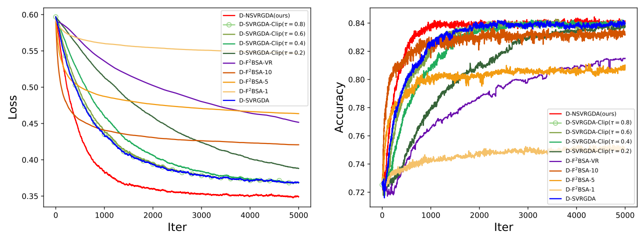
5.2 Experimental Setup

All experiments are conducted in a decentralized environment where workers are connected by a fixed communication graph. Unless otherwise stated, we use the same communication topology and mixing matrix across all methods and models.

For the SVM experiments, we use a mini-batch size of 32 and a hidden feature dimension of 20 (for the corresponding feature transformation layer when applicable). The learning rate is set to 0.001, with $\rho = 0.1$ and momentum 0.9. The target pruning rate is fixed at approximately 20%, meaning that about one fifth of the weights are removed by the learned mask at convergence.

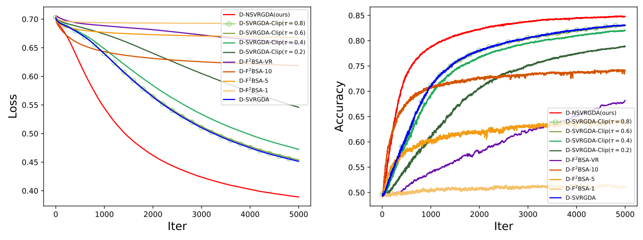
For the RNN experiments on Sent140, we use a batch size of 16 and a hidden dimension of 128. The learning rate, ρ, and momentum are kept the same as in the SVM setting (learning rate 0.001, $\rho = 0.1$, momentum 0.9), and we again target a pruning rate of about 20%.

For the CNN experiments with ResNet-18 on CIFAR-10, we use a batch size of 16 and train the model in a decentralized setting with 8 workers connected in a ring topology. The learning rate is set to 0.001, with $\rho = 0.1$ and momentum 0.9, and we again target a pruning rate of about 20%. As in the other settings, we apply unstructured pruning to the convolutional and fully connected layers via the same bi-level optimization scheme described in Section 4, alternating



(a) Upper loss on a9a (b) Test accuracy on a9a

Figure 1. SVM results on a9a (lr = 0.001, ϵ = 0.1).



(a) Upper loss on IMDB (b) Test accuracy on IMDB

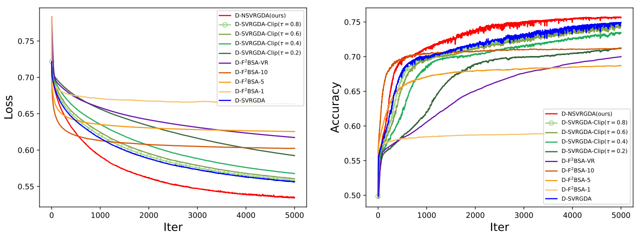
Figure 2. SVM results on IMDB (lr = 0.001, ϵ = 0.1).

between inner-loop updates of the model parameters and outer-loop updates of the pruning mask.

5.3 Results

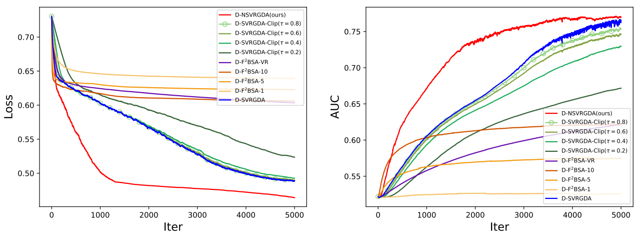
We compare our method against several strong decentralized optimization baselines, including D-SVRGDA, a variance-reduced decentralized gradient method used as a non-pruned reference, and D-SVRGDA-Clip, a clipped variant with thresholds $\tau \in 0.8, 0.6, 0.4, 0.2$ that induces sparsity by limiting the magnitude of communicated updates. We also include the D-Φ^2BSA family of bi-level stochastic approximation algorithms—D-Φ^2BSA-VR, D-Φ^2BSA-10, D-Φ^2BSA-5, and D-Φ^2BSA-1—which differ in the number of inner steps and in whether variance reduction is applied in the outer loop. All baselines use the same decentralized communication graph and, whenever applicable, similar learning rate and batch-size configurations to ensure a fair comparison with our proposed method.

We report results in terms of training/upper loss, test accuracy or AUC, and the sparsity level of the final model. The SVM results on the a9a, IMDB, and covtype datasets are shown in Figures 1, 2, and 3, respectively. The RNN results on Sent140 are shown in Figure 4, and the CNN results on CIFAR-10 are reported in Figure 5. For the CNN experiments, in order to obtain stable training and focus on the effect of pruning in the decentralized setting, we follow existing practice and initialize the network from a pretrained ResNet-18 model rather than training it from scratch.



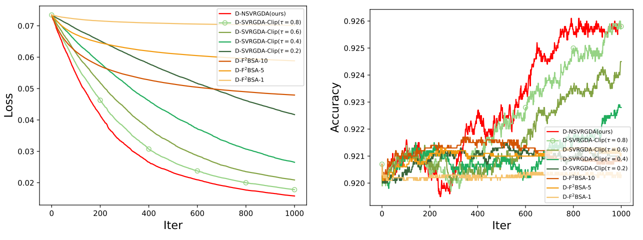
(a) Upper loss on covtype (b) Test accuracy on covtype

Figure 3. SVM results on covtype (lr = 0.001, ϵ = 0.1).



(a) Upper loss on Sent140 (b) Test AUC on Sent140

Figure 4. RNN results on Sent140 (lr = 0.001, ϵ = 0.1).



(a) Upper loss on CIFAR-10 (b) Test accuracy on CIFAR-10

Figure 5. ResNet-18 results on CIFAR-10 (lr = 0.001, ϵ = 0.1).

Across all model families, our method consistently achieves the best or near-best performance. For the SVM benchmarks (a9a, IMDB, and covtype), our method achieves the lowest upper loss and the highest or near-highest test accuracy throughout training, while maintaining a pruning rate of around 20%, yielding a significantly smaller model. On Sent140, our RNN experiments show that the proposed method exhibits a much faster decrease in upper loss and converges to a higher test AUC than all baselines, indicating that the bi-level pruning does not harm, and may even stabilize, optimization in sequence models. On CIFAR-10 with ResNet-18, our method reaches competitive or superior test accuracy compared to non-pruned baselines, while removing a substantial fraction of weights via unstructured pruning, demonstrating that the proposed framework can scale to deeper architectures without sacrificing accuracy.

Overall, these results suggest that the proposed decentralized bi-level pruning framework can simultaneously improve communication efficiency and maintain (or even enhance) model performance across a diverse set of architectures and datasets.

6 Conclusions

In this work, we proposed a bi-level optimization framework for model pruning in decentralized environments. By explicitly separating the pruning mask (upper level) from the model parameters (lower level) and coupling them through a bi-level objective, our method provides a principled alternative to heuristic pruning strategies commonly used in distributed learning.

We instantiated the framework on linear SVMs, RNNs, and a CNN (ResNet-18), and evaluated it across several benchmarks, including a9a, IMDB, covtype, Sent140, and CIFAR-10. Empirically, our approach achieves faster convergence, lower training or upper-level loss, and higher test accuracy or AUC compared with strong decentralized baselines, while maintaining a nontrivial pruning rate (around 20%) that significantly reduces the effective model size. These results indicate that bi-level pruning can simultaneously improve communication efficiency and preserve, or even enhance, predictive performance in decentralized settings.

At the same time, our preliminary CNN experiments suggest that larger and deeper architectures remain more sensitive to optimization and initialization, pointing to several promising directions for future work, including more robust hypergradient approximations, adaptive pruning schedules, and extensions to additional model families and graph topologies.

Acknowledgments

For privacy and academic integrity considerations, and because we plan to conduct further research and attempt to publish this work, we respectfully request that this report and its accompanying materials not be posted or shared publicly online.

References

[1] Davis Blalock et al. 2020. What is the state of neural network pruning? *arXiv preprint arXiv:2003.03033* (2020).

[2] Benoît Colson et al. 2007. An overview of bilevel optimization. *Annals of Operations Research* (2007).

[3] Luca Franceschi et al. 2018. Bilevel programming for hyperparameter optimization. *ICML* (2018).

[4] Jonathan Frankle and Michael Carbin. 2019. The lottery ticket hypothesis. *ICLR* (2019).

[5] Song Han, Huizi Mao, and William Dally. 2015. Deep compression: Compressing deep neural networks with pruning. *arXiv preprint arXiv:1510.00149* (2015).

[6] Song Han, Jeff Pool, John Tran, and William Dally. 2015. Learning both weights and connections for efficient neural networks. *NeurIPS* (2015).

[7] Peter Kairouz et al. 2021. Advances and open problems in federated learning. *Foundations and Trends in Machine Learning* (2021).

[8] Christos Louizos et al. 2018. Learning sparse neural networks through l_0 regularization. *ICLR* (2018).

[9] Yihua Zhang et al. 2023. Advancing model pruning via bi-level optimization. *NeurIPS* (2023).