PRUNING DEEP NEURAL NETWORKS FROM A SPARSITY PERSPECTIVE

Anonymous authors
Paper under double-blind review

ABSTRACT

In recent years, deep network pruning has attracted significant attention in order to enable the rapid deployment of AI into small devices with computation and memory constraints. Pruning is often achieved by dropping redundant weights, neurons, or layers of a deep network while attempting to retain a comparable test performance. Many deep pruning algorithms have been proposed with impressive empirical success. However, existing approaches lack a quantifiable measure to estimate the compressibility of a sub-network during each pruning iteration and thus may under-prune or over-prune the model. In this work, we propose PQ Index (PQI) to measure the potential compressibility of deep neural networks and use this to develop a Sparsity-informed Adaptive Pruning (SAP) algorithm. Our extensive experiments corroborate the hypothesis that for a generic pruning procedure, PQI decreases first when a large model is being effectively regularized and then increases when its compressibility reaches a limit that appears to correspond to the beginning of underfitting. Subsequently, PQI decreases again when the model collapse and significant deterioration in the performance of the model start to occur. Additionally, our experiments demonstrate that the proposed adaptive pruning algorithm with proper choice of hyper-parameters is superior to the iterative pruning algorithms such as the lottery ticket-based pruning methods, in terms of both compression efficiency and robustness.

1 INTRODUCTION

Over-parameterized deep neural networks have been applied with enormous success in a variety of fields, including computer vision (Krizhevsky et al., 2012; He et al., 2016b; Redmon et al., 2016), natural language processing (Devlin et al., 2018; Radford et al., 2018), and audio signal processing (Oord et al., 2016; Schneider et al., 2019). These deep neural networks have significantly expanded in size. For example, LeNet-5 (LeCun et al., 1998) (1998; image classification) has 60 thousand parameters whereas GPT-3 (Brown et al., 2020) (2020; language modeling) has 175 billion parameters. This rapid growth in size has necessitated the deployment of a vast amount of computation, storage, and energy resources. Due to hardware constraints, these enormous model sizes may be a barrier to deployment in some edge devices such as mobile phones and virtual assistants. This has greatly increased interest in deep neural network compression/pruning. To this end, various researchers have developed empirical methods of building much simpler networks with similar performance based on pre-trained networks (Han et al., 2015; Frankle & Carbin, 2018). For example, Han et al. (2015) demonstrated that AlexNet (Krizhevsky et al., 2017) could be compressed to retain only 3% of the original parameters on the ImageNet dataset without impacting classification accuracy.

An important topic of interest is the determination of limits of network pruning. An overly pruned model may not have enough expressivity for the underlying task, which may lead to significant performance deterioration. Existing methods generally monitor the prediction performance on a validation dataset and terminate pruning when the performance falls below a pre-specified threshold. Nevertheless, a quantifiable measure for estimating the compressibility of a sub-network during each pruning iteration is desired. Such quantification of compressibility can lead to the discovery of the most parsimonious sub-networks without performance degradation.

In this work, we connect the compressibility and performance of a neural network to its sparsity. In a highly over-parameterized network, one popular assumption is that the relatively small weights are

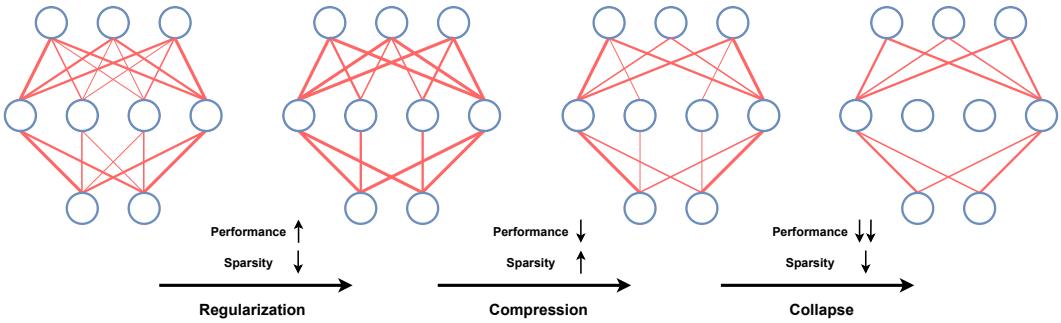

Figure 1: An illustration of our hypothesis on the relationship between sparsity and compressibility of neural networks. The width of connections denotes the magnitude of model parameters.

considered redundant or non-influential and may be pruned without impacting the performance. Let us consider the sparsity of a non-negative vector $w = [w_1, \ldots, w_d]$, since sparsity is related only to the magnitudes of entries. Suppose $S(w)$ is a sparsity measure, and a larger value indicates higher sparsity. Hurley & Rickard (2009) summarize six properties that an ideal sparsity measure should have, originally proposed in economics (Dalton, 1920; Rickard & Fallon, 2004). They are

(D1) Robin Hood. For any $w_i > w_j$ and $\alpha \in (0, (w_i - w_j)/2)$, we have $S([w_1, \ldots, w_i - \alpha, \ldots, w_j + \alpha, \ldots, w_d]) < S(w)$.

(D2) Scaling. $S_q(\alpha w) = S_q(w)$ for any $\alpha > 0$.

(D3) Rising Tide. $S_q(w + \alpha) < S_q(w)$ for any $\alpha > 0$ and w_i not all the same.

(D4) Cloning. $S(w) = S([w, w])$.

(P1) Bill Gates. For any $i = 1, \ldots, d$, there exists $\beta_i > 0$ such that for any $\alpha > 0$ we have

$$S([w_1, \ldots, w_i + \beta_i + \alpha, \ldots, w_d]) > S([w_1, \ldots, w_i + \beta_i, \ldots, w_d]).$$

(P2) Babies. $S([w_1, \ldots, w_d, 0]) > S(w)$ for any non-zero w.

Hurley & Rickard (2009) point out that only Gini index satisfies all six criteria among a comprehensive list of sparsity measures. In this work, we propose a measure of sparsity named PQ Index (PQI). To the best of our knowledge, PQI is the first measure related to the norm of a vector that satisfies all the six properties above. Therefore, PQI is an ideal indicator of vector sparsity and is of its own interest. We suggest using PQI to infer the compressibility of neural networks. Furthermore, we discover the relationship between the performance and sparsity of iteratively pruned models as illustrated in Figure 1. Our hypothesis is that *for a generic pruning procedure, the sparsity will first decrease when a large model is being effectively regularized, then increase when its compressibility reaches a limit that corresponds to the start of underfitting, and finally decrease when the model collapse occurs, i.e., the model performance significantly deteriorates.*

Our intuition is that the pruning will first remove redundant parameters. As a result, the sparsity of model parameters will decrease and the performance may be improved due to regularization. When the model is further compressed, part of the model parameters will become smaller when the model converges. Thus, the sparsity will increase and the performance will moderately decrease. Finally, when the model collapse starts to occur, all attenuated parameters are removed and the remaining parameters become crucial to maintain the performance. Therefore, the sparsity will decrease and performance will significantly deteriorate. Our extensive experiments on pruning algorithms corroborate the hypothesis. Consequently, PQI can infer whether a model is inherently compressible. Motivated by this discovery, we also propose the Sparsity-informed Adaptive Pruning (SAP) algorithm, which can compress more efficiently and robustly compared with iterative pruning algorithms such as the lottery ticket-based pruning methods. Overall, our work presents a new understanding of the inherent structures of deep neural networks for model compression. Our main contributions are summarized below.

1. We propose a new notion of sparsity for vectors named PQ Index (PQI), with a larger value indicating higher sparsity. We prove that PQI meets all six properties proposed by (Dalton, 1920; Rickard & Fallon, 2004), which capture the principles a sparsity measure should obey. Among 15 commonly used sparsity measures, the only other measure satisfying all properties is Gini Index (Hurley & Rickard, 2009). Thus, norm-based PQI is an ideal sparsity/equity measure and may be of independent interest to many areas, e.g., signal processing and economics.
2. We develop a new perspective on the compressibility of neural networks. In particular, we measure the sparsity of pruned models by PQI and postulate the above hypothesis on the relationship between sparsity and compressibility of neural networks.
3. Motivated by our proposed PQI and hypothesis, we further develop a Sparsity-informed Adaptive Pruning (SAP) algorithm that uses PQI to choose the pruning ratio adaptively. In particular, the pruning ratio at each iteration is decided based on a PQI-related inequality. In contrast, Gini Index does not have such implications for the pruning ratio.
4. We conduct extensive experiments to measure the sparsity of pruned models and corroborate our hypothesis. Our experimental results also demonstrate that SAP with proper choice of hyper-parameters can compress more efficiently and robustly compared with iterative pruning algorithms such as the lottery ticket-based pruning methods.

2 Related Work

Model compression The goal of model compression is to find a smaller model that has comparable performance to the original model. A smaller model saves storage and computation resources, boosts training, and facilitates the deployment of the model to devices with limited capacities, such as mobile phones and virtual assistants. Therefore, model compression is vital for deploying deep neural networks with millions or even billions of parameters. Various model compression methods have been proposed for neural networks. Among them, pruning is one of the most popular and effective approach (LeCun et al., 1989; Hagiwara, 1993; Han et al., 2015; Hu et al., 2016; Luo et al., 2017; Frankle & Carbin, 2018; Lee et al., 2018; He et al., 2017). The idea of pruning is the sparsity assumption that many redundant or non-influential neuron connections exist in an over-parameterized model. Thus, we can remove those connections (e.g., weights, neurons, or neuron-like structures such as layers) without sacrificing much test accuracy. Two critical components of pruning algorithms are a pruning criterion that decides which connection to be pruned and a stop criterion that determines when to stop pruning and thus prevent underfitting and model collapse. There are many pruning criteria motivated by different interpretations of redundancy. A widely-used criterion removes the parameters with the smallest magnitudes, assuming that they are less important (Hagiwara, 1993; Han et al., 2015). Besides magnitude-based pruning, one may prune the parameters based on their sensitivity or contribution to the network output (LeCun et al., 1989; Lee et al., 2018; Hu et al., 2016; Soltani et al., 2021). As for the stop criterion, the common choice is validation: to stop pruning once the test accuracy on a validation dataset falls below a given threshold.

While pruning is a post-processing method that requires a pre-trained model, there are also pre-processing and in-processing methods based on the sparsity assumption. For example, one can add explicit sparse constraints on the network, such as forcing the parameters to have a low-rank structure and sharing weights. Alternatively, one can implicitly force the trained model to be sparse, such as adding a sparsity penalty (e.g., ℓ_1-norm) to the parameters. In contrast to those sparsity-based compression methods, which find a sub-network of the original one, researchers also propose compressing the model by finding a smaller model with different architecture. The efforts include knowledge distillation and architecture search. We refer the reader to Hoefler et al. (2021) for a comprehensive survey of model compression.

Theory of model compression In practice, the compressibility of a model depends on the network architecture and learning task. The pruning usually involves a lot of ad hoc hyper-parameter fine-tuning. Thus, an understanding of model compressibility is urgently needed. There have been some recent works to show the existence of or find a sub-network with guaranteed performance. Arora et al. (2018) show that a model is more compressible if it is more stable to the noisy inputs and provides a generalization error bound of the pruned model. Yang et al. (2022) propose a backward pruning algorithm inspired by approximating functions using ℓ_q-norm (Wang et al., 2014), and quantify its

generalization error. Baykal et al. (2018); Mussay et al. (2019) utilize the concept of coreset to prove the existence of a pruned network with similar performance. The main idea is to sample the parameters based on their importance, and thus selected parameters could preserve the output of the original network. Ye et al. (2020) propose a greedy selection algorithm to reconstruct a network and bound the generalization error for two-layer neural networks. Our work develops a new perspective on the compressibility of neural networks by directly measuring the sparsity of pruned models to reveal the relationship between the sparsity and compressibility of neural networks.

Sparsity measure Sparsity is a crucial concept in many fundamental fields such as statistics and signal processing (Tibshirani, 1996; Donoho, 2006; Akçakaya & Tarokh, 2008). Intuitively, sparsity means that the most energy is concentrated in a few elements. For example, a widely-used assumption in high-dimensional machine learning is that the model has an underlying sparse representation. Various sparsity measures have been proposed in the literature from different angles. One kind of sparsity measure originates from sociology and economics. For example, the well-known Gini Index (Gini, 1912) can measure the inequality in a population's wealth or welfare distribution. A highly wealth-concentrated population forms a sparse vector if the vector consists of the wealth of each person. In addition, to measure the diversity in a group, entropy-based measures like Shannon entropy and Gaussian entropy are often used (Jost, 2006). Another kind of sparsity measure has been studied in mathematics and engineering for a long time. A classic measure is the hard sparsity, also known as ℓ_0-norm, which is the number of non-zero elements in w. A small hard sparsity implies that only a few vector elements are active or effective. However, a slight change in the zero-valued element may cause a significant increase in the hard sparsity, which can be undesirable. Thus, its relaxations such as ℓ_p-norm ($0 < p \leq 1$) are also widely used. For example, ℓ_1-norm-based constraints or penalties are used for function approximation (Barron, 1993), model regularization and variable selection (Tibshirani, 1996; Chen et al., 2001). Our work proposes the first measure of sparsity related to vector norms that satisfies all the properties shared by the Gini Index (Hurley & Rickard, 2009) and an adaptive pruning algorithm based on our proposed measure of sparsity.

3 Pruning with PQ Index

3.1 PQ Index

We will prove all the six properties (D1)-(D4) and (P1), (P2), which are mentioned in the introduction, hold for our proposed PQ Index (PQI).

Definition 1 (PQ Index)**.** *For any $0 < p < q$, the PQ Index of a non-zero vector $w \in \mathbb{R}^d$ is*

$$\mathrm{I}_{p,q}(w) = 1 - d^{\frac{1}{q}-\frac{1}{p}} \frac{\|w\|_p}{\|w\|_q}, \tag{1}$$

where $\|w\|_p = (\sum_{i=1}^{d} |w_i|^p)^{1/p}$ is the ℓ_p-norm of w for any $p > 0$. For simplicity, we will use $\mathrm{I}(w)$ and drop the dependency on p and q when the context is clear.

Theorem 1. *We have $0 \leq \mathrm{I}_{p,q}(w) \leq 1 - d^{\frac{1}{q}-\frac{1}{p}}$, and a larger $\mathrm{I}_{p,q}(w)$ indicates a sparser vector. Furthermore, $\mathrm{I}_{p,q}(w)$ satisfies all the six properties (D1)-(D4) and (P1), (P2).*

Remark 1 (Sanity check)**.** *For the densest or most equal situation, we have $w_i = c$ for $i = 1, \ldots, d$, where c is a non-zero constant. It can be verified that $\mathrm{I}_{p,q}(w) = 0$. In contrast, the sparsest or most unequal case is that w_i's are all zeros except one of them, and corresponding $\mathrm{I}_{p,q}(w) = 1 - d^{\frac{1}{q}-\frac{1}{p}}$. Note that $\mathrm{I}(w)$ for an all-zero vector is not defined. From the perspective of the number of important elements, an all-zero vector is sparse; however, it is dense from the aspect of energy distribution.*

Remark 2 (Insights)**.** *The form of $\mathrm{I}_{p,q}$ is not a random thought but inherently driven by properties (D1)-(D4). Why do we need the ratio of two norms? It is essentially decided by the requirement of (D2) Scaling. If $S(w)$ involves only a single norm, then $S(w)$ is not scale-invariant. However, since ℓ_r-norm is homogeneous for all $r > 0$, the ratio of two norms is inherently scale-invariant. Why is there an additional scaling constant $d^{\frac{1}{q}-\frac{1}{p}}$? This is necessary to satisfy (D4) Cloning. Inspired by the well-known Root Mean Squared Error (RMSE), we found out that the additional scaling constant is the correct term to help $\mathrm{I}_{p,q}$ be independent of the vector length. It is essentially appealing for comparing the sparsity of neural networks with different model parameters. Why do we require $p < q$? We find it plays a central role in meeting (D1) and (D3). The insight is that $\|w\|_p$ decreases faster than $\|w\|_q$ when a vector becomes sparser, thus guaranteeing a larger PQ Index.*

Theorem 2 (PQI-bound on pruning)**.** *Let M_r denote the set of r indices of w with the largest magnitudes, and η_r be the smallest value such that $\sum_{i \notin M_r} |w_i|^p \leq \eta_r \sum_{i \in M_r} |w_i|^p$. Then, we have*

$$r \geq d(1+\eta_r)^{-q/(q-p)}[1-\mathrm{I}(w)]^{\frac{qp}{q-p}}. \tag{2}$$

Remark 3. *The PQI-bound is inspired by Yang et al. (2022) that proposed to use $\|w\|_1/\|w\|_q, q \in (0,1)$ as a measure of sparsity. We use similar techniques to derive the bound based on our proposed PQ Index. It is worth mentioning that $\|w\|_1/\|w\|_q$ does not satisfy properties (D2), (D4), (P1), and (P4), which an ideal sparsity measure should have Hurley & Rickard (2009). Consequently, we cannot use it to compare the sparsity of models of different sizes. The merit of the PQI-bound is that it applies to iterative pruning of models that involve different numbers of model parameters.*

Recall that the pruning is based on the assumption that parameters with small magnitudes are removable. Therefore, suppose we know $\mathrm{I}(w)$ and η_r, then we immediately have a lower bound for the retaining ratio of the pruning from Theorem 2. Thus, we can adaptively choose the pruning ratio based on $\mathrm{I}(w)$ and η_r, which inspires our Sparsity-informed Adaptive Pruning (SAP) algorithm in the following subsection. In practice, η_r is unavailable before we decide the pruning ratio. Therefore, we treat it as a hyper-parameter in our experiments. Since η_r is non-increasing with respect to r, a larger η_r means that we assume the model is more compressible and leads to a higher pruning ratio. Experiments show that it is safe to choose $\eta_r = 0$.

3.2 Sparsity-informed Adaptive Pruning

In this section, we introduce the Sparsity-informed Adaptive Pruning (SAP) algorithm as illustrated in Algorithm 1. Our algorithm is based on the well-known lottery ticket pruning method (Frankle & Carbin, 2018). The lottery ticket pruning algorithm proposes to prune and retrain the model iteratively. Compared with one shot pruning algorithm, which does not retrain the model at each pruning iteration, the lottery ticket pruning algorithm produces pruned models of better performance with the same percent of remaining model parameters. However, both methods use a fixed pruning ratio P at each pruning iteration. As a result, they may under-prune or over-prune the model at earlier or later pruning iterations, respectively. The under-pruned models with spare compressibility require more pruning iterations and computation resources to obtain the smallest neural networks with satisfactory performance. The over-pruned model suffers from underfitting due to insufficient compressibility to maintain the desired performance. Therefore, we propose SAP to adaptively determine the number of pruned parameters at each iteration based on the PQI-bound derived in Formula 2. Furthermore, we introduce two additional hyper-parameters, the scaling factor γ and the maximum pruning ratio β, to make our algorithm further flexible and applicable.

Algorithm 1 Sparsity-informed Adaptive Pruning (SAP)

Input: model parameters w, mask m, norm $0 < p < q$, compression hyper-parameter η_r, scaling factor γ, maximum pruning ratio β, number of epochs E, and number of pruning iterations T.

Randomly generate model parameters w_{init}
Initialize mask m_0 with all ones
for each pruning iteration $t = 0, 1, 2, \ldots T$ **do**
  Initialize model parameters $\tilde{w}_t = w_{\text{init}} \odot m_t$
  Compute the number of model parameters $d_t = |m_t|$
  Train the model parameters $\tilde{w}_t$ with m_t for E epochs and arrive at w_t
  Compute PQ Index $\mathrm{I}(w_t) = 1 - d_t^{\frac{1}{q}-\frac{1}{p}} \frac{\|w_t\|_p}{\|w_t\|_q}$
  Compute the lower bound of the number of retained model parameters
  $r_t = d_t(1+\eta_r)^{-q/(q-p)}[1-\mathrm{I}(w_t)]^{\frac{qp}{q-p}}$
  Compute the number of pruned model parameters
  $c_t = \lfloor d_t \cdot \min(\gamma(1-\frac{r_t}{d_t}), \beta) \rfloor$
  Prune c_t model parameters with the smallest magnitude based on w_t and m_t
  Create new mask m_{t+1}
end
Output: The pruned model parameters w_T and mask m_T.

Next, we walk through our SAP algorithm. Before the pruning starts, we randomly generate model parameters w_{init}. We will use w_{init} to initialize retained model parameters at each pruning iterations.

The lottery ticket pruning algorithm shows that the performance of retraining the subnetworks from w_{init} is better than from scratch. Then, we initialize the mask m_0 with all ones. Suppose we have T number of pruning iterations. For each pruning iteration $t = 0, 1, 2, \ldots T$, we first initialize the model parameters $\tilde{w}_t$ and compute the number of model parameters d_t as follows

$$\tilde{w}_t = w_{\text{init}} \odot m_t, \quad d_t = |m_t|, \tag{3}$$

where $\odot$ is the Hadamard product. After training the model parameters $\tilde{w}_t$ with m_t by freezing the gradient for E epoch, we arrive at trained model parameters w_t. Upon this point, our algorithm has no difference from the classical lottery ticket pruning algorithm. The lottery ticket pruning algorithm will then prune $d \cdot P$ parameters from m_t according to the magnitude of w_t and finally create new mask m_{t+1}.

After arriving at w_t, our proposed SAP will compute the PQ Index, denoted by $\text{I}(w_t)$, and the lower bound of the number of retrained model parameters, denoted by r_t, as follows

$$\text{I}(w_t) = 1 - d_t^{\frac{1}{q}-\frac{1}{p}} \frac{\|w_t\|_p}{\|w_t\|_q}, \qquad r_t = d_t(1+\eta_r)^{-q/(q-p)}[1-\text{I}(w_t)]^{\frac{qp}{q-p}}. \tag{4}$$

Then, we compute the number of pruned model parameters $c_t = \lfloor d_t \cdot \min(\gamma(1-\frac{r_t}{d_t}), \beta) \rfloor$. Here, we introduce γ to accelerate or decelerate the pruning at initial pruning iterations. Specifically, $\gamma > 1$ or < 1 encourages the pruning ratio to be larger or smaller than the pruning ratio derived from the PQI-bound, respectively. As the retraining of pruned models is time-consuming, it is appealing to efficiently obtain the smallest pruned model with satisfactory performance for a small number of pruning iterations. In addition, we introduce the maximum pruning ratio β to avoid excessive pruning because we will compress more than the PQI-bound if $\gamma > 1$ and may completely prune all model parameters. If we set $\gamma = 1$, β can be safely omitted. In our experiments, we set $\beta = 0.9$ only to provide minimum protection from excessive pruning at each pruning iteration. Finally, we prune c_t model parameters with the smallest magnitude based on w_t and m_t, and finally create new mask m_{t+1} for the next pruning iteration.

4 Experimental Studies

4.1 Experimental Setup

We conduct experiments with FashionMNIST (Xiao et al., 2017), CIFAR10, CIFAR100 (Krizhevsky et al., 2009) datasets. Our backbone models are Linear, Multi-Layer Perceptron (MLP), Convolutional Neural Network (CNN), ResNet18 (He et al., 2016a), and Wide ResNet28x8 (WResNet28x8) (Zagoruyko & Komodakis, 2016). We run experiments for $T = 30$ pruning iterations with Linear, MLP, and CNN, and $T = 15$ pruning iterations with ResNet18. We compare the proposed SAP with two baselines, including 'One Shot' and 'Lottery Ticket' (Frankle & Carbin, 2018) pruning algorithms. The difference between 'One Shot' and 'Lottery Ticket' pruning algorithms is that 'One Shot' prunes $d \cdot P$ model parameters at each pruning iteration from w_0 instead of w_t. We have $P = 0.2$ throughout our experiments. We compare the proposed PQ Index ($p = 0.5$, $q = 1.0$) with the well-known Gini Index (Gini, 1912) to validate its effectiveness in evaluating sparsity. Furthermore, we perform pruning on various pruning scopes, including 'Neuron-wise Pruning,' 'Layer-wise Pruning,' and 'Global Pruning.' In particular, 'Global Pruning' gather all model parameters as a vector for pruning, while 'Neuron-wise Pruning' and 'Layer-wise Pruning' prune each neuron and layer of model parameters separately. The number of neurons at each layer is equal to the output size of that layer. For example, 'One Shot' and 'Lottery Ticket methods with 'Neuron-wise Pruning' prune $d_i \cdot P$ model parameters of each neuron, where d_i refers to the size of each neuron. Similarly, SAP computes the PQ Index and the number of pruned model parameters for neuron i. Details of the model architecture and learning hyper-parameters are included in the Appendix. We conducted four random experiments with different seeds, and the standard deviation is shown in the error bar of all figures. Further experimental results can be found in the Appendix.

4.2 Experimental Results

Retrained models We demonstrate the results of retrained models ($w_t \odot m_t$) at each pruning iteration in Figure 2(a) and 3(a). In particular, we illustrate the performance, percent of remaining weights, PQ Index, and Gini Index at each pruning iteration. In these experiments, η_r and γ are set to 0 and 1 to prevent interference. The 'Lottery Ticket' method outperforms 'One Shot' as expected. SAP

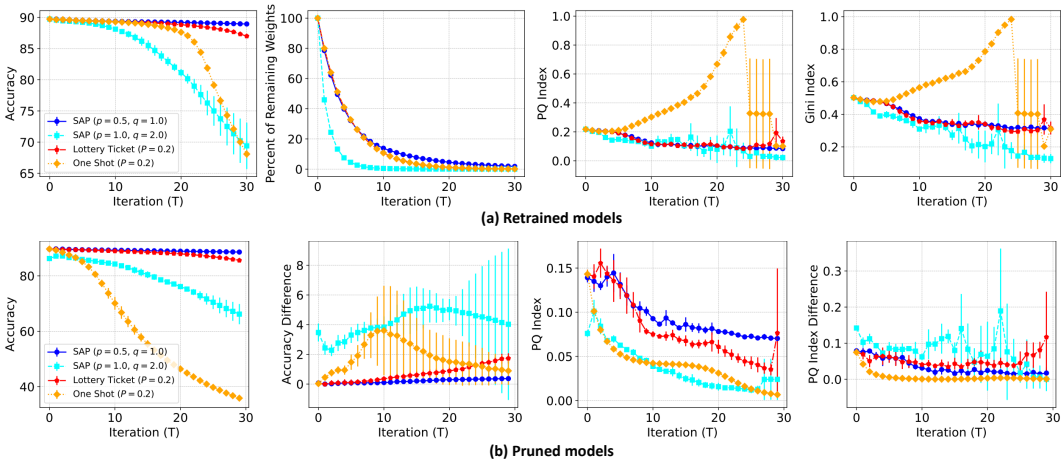

Figure 2: Results of (a) retrained and (b) pruned models at each pruning iteration for 'Global Pruning' with FashionMNIST and MLP, where (a) is obtained from the models after retraining and (b) is directly from those after pruning.

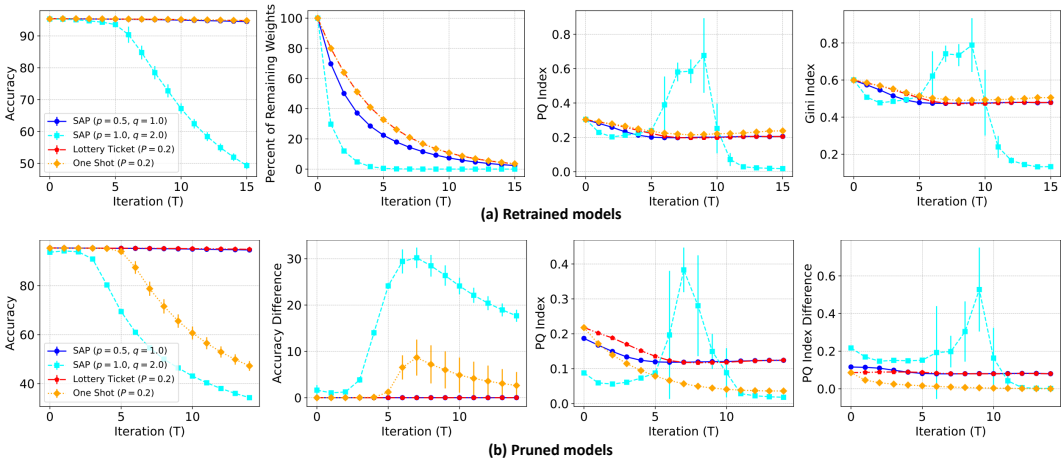

Figure 3: Results of (a) retrained and (b) pruned models at each pruning iteration for 'Global Pruning' with CIFAR10 and ResNet18.

($p = 1.0$, $q = 2.0$) compresses more aggressively than SAP ($p = 0.5$, $q = 1.0$). Recall that SAP adaptively adjusts the pruning ratio based on the sparsity of models, while 'One Shot' and 'Lottery Ticket' have a fixed pruning ratio at each pruning iteration. In Figure 2(a), SAP ($p = 1.0$, $q = 2.0$) at around $T = 10$ achieves similar performance as 'Lottery Ticket' at around $T = 25$. Furthermore, SAP ($p = 0.5$, $q = 1.0$) and 'Lottery Ticket' have similar pruning ratio before $T = 10$, but SAP ($p = 0.5$, $q = 1.0$) adaptively lowers the pruning ratio and thus prevents the performance from deteriorating like 'Lottery Ticket' does after $T = 25$. In Figure 3(a), we observe similar fast pruning phenomenon of SAP ($p = 1.0$, $q = 2.0$). Meanwhile, SAP ($p = 0.5$, $q = 1.0$) performs similar to 'Lottery Ticket' but prunes more than 'Lottery Ticket' does. Consequently, by carefully choosing p and q, SAP can provide more efficient and robust pruning. As for the sparsity of retrained models, the result of the PQ Index is aligned with the Gini Index, which experimentally demonstrates that the PQ Index can effectively measure the sparsity of model parameters. Furthermore, the dynamics of the sparsity also corroborates our hypothesis, as shown by 'One Shot' in Figure 2(a) and SAP ($p = 1.0$, $q = 2.0$) in Figure 3(a). The results show that an ideal pruning procedure should avoid a rapid increase in sparsity.

Pruned models We demonstrate the results of pruned models at each pruning iteration in Figure 2(b) and 3(b). In particular, we illustrate the performance, performance difference, PQ Index, and PQ Index difference at each pruning iteration. The performance and PQ Index are computed directly

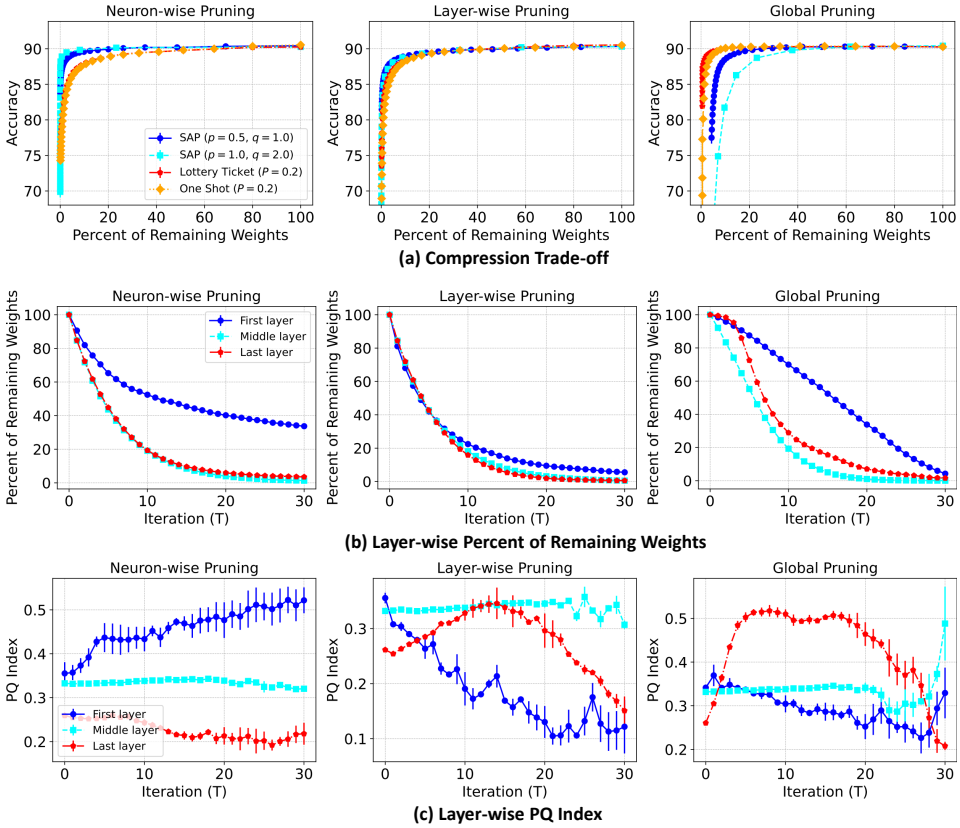

Figure 4: Results of various pruning scopes regarding (a) compression trade-off, (b) layer-wise percent of remaining weights, and (c) layer-wise PQ Index for CIFAR10 and CNN. (b, c) are performed with SAP ($p = 0.5$, $q = 1.0$).

from the pruned models without retraining. The performance and PQ Index difference is between the retrained ($w_0 \odot m_t$ for 'One Shot' and $w_t \odot m_t$ for 'Lottery Ticket' and SAP) and pruned models ($w_0 \odot m_{t+1}$ for 'One Shot' and $w_t \odot m_{t+1}$ for 'Lottery Ticket' and SAP). The results of performance difference show that the performance of pruned models without retraining from SAP can perform close to that of retrained models. Furthermore, the sparsity of pruned models shows that iterative pruning generally decreases the sparsity of pruned models. Meanwhile, the sparsity difference of pruned models provides a sanity check by showing that pruning at each pruning iteration decreases the sparsity of retrained models.

Pruning scopes We demonstrate various pruning scopes regarding compression trade-off, layer-wise percent of remaining weights, and layer-wise PQ Index in Figure 4. The results of the compression trade-off show that SAP with 'Global Pruning' may perform worse than 'One Shot' and 'Lottery Ticket' when the percent of remaining weights is small. As illustrated in the 'Global Pruning' of Figure 4(b), the first layer has not been pruned enough. It is because SAP with 'Global Pruning' measures the sparsity of all model parameters in a vector, and the magnitude of the parameters of the first layer and other layers may not be at the same scale. As a result, the parameters of the first layer will not be pruned until late pruning iterations. However, SAP with 'Neuron-wise Pruning' and 'Layer-wise Pruning' perform better than 'One Shot' and 'Lottery Ticket.' SAP can adaptively adjust the pruning ratio of each neuron and layer, but 'One Shot' and 'Lottery Ticket' may over-prune specific neurons and layers because they adopt a fixed pruning ratio. Interestingly, the parameters of the first layer are pruned more aggressively by 'Neuron-wise Pruning' than 'Global Pruning' in the early pruning iterations. However, they are not pruned by 'Neuron-wise Pruning' in the late pruning iterations, while 'Global Pruning' still prunes them aggressively. It aligns with the intuition that the initial layers of CNN are more important to maintain the performance, e.g., Gale et al. (2019) observed that the first layer was often more important to model quality and pruned less than other layers. Furthermore, the PQ Index of 'Neuron-wise Pruning' is also more stable than the other two

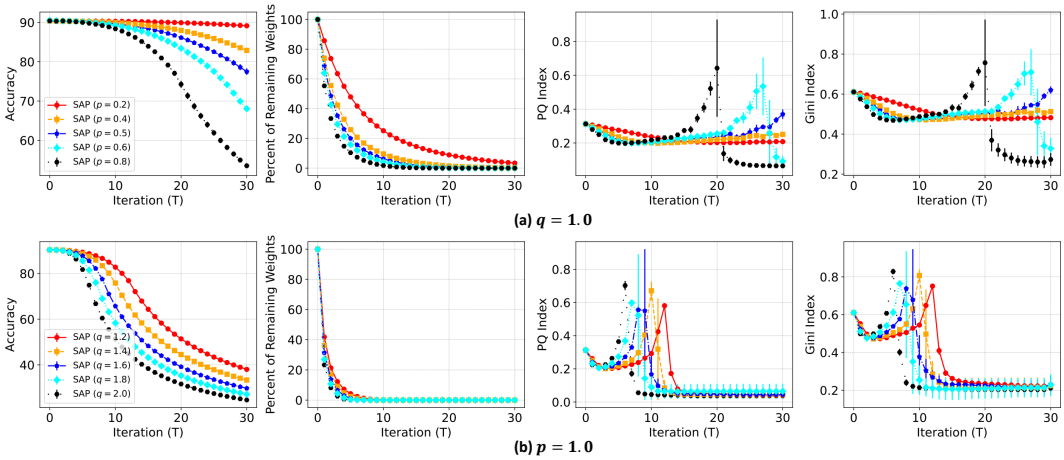

Figure 5: Ablation studies of p and q for global pruning with CIFAR10 and CNN.

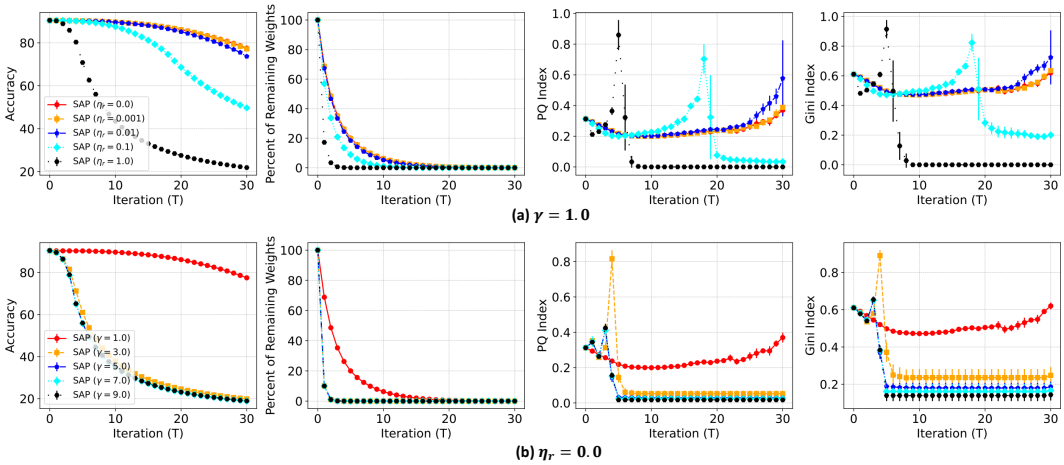

Figure 6: Ablation studies of η_r and γ for global pruning with CIFAR10 and CNN.

pruning scopes, which indicates that 'Neuron-wise Pruning' is more appropriate for SAP, as the PQ Index is computed more precisely.

Ablation studies We demonstrate ablation studies of p and q in Figure 5. In Figure 5(a), we fix $q = 1.0$ and study the effect of p. The results show that SAP prunes more aggressively when $q = 1.0$ and p is close to q. In Figure 5(b), we fix $p = 1.0$ and study the effect of q. The results show that SAP prunes more aggressively when $p = 1.0$ and q is distant from p. We demonstrate ablation studies of η_r and γ in Figure 6. In Figure 6(a), we fix $\gamma = 1.0$ and study the effect of η_r. The results show that SAP prunes more aggressively when $\eta_r > 0$. In Figure 6(b), we fix $\eta_r = 0.0$ and study the effect of γ. The results show that SAP prunes more aggressively when $\gamma > 1$. Interestingly, the performance of our results roughly follows a logistic decay model due to the adaptive pruning ratio, and the inflection point corresponds to the peak of the sparsity measure. Moreover, the dynamics of the sparsity measure of SAP with various ablation studies also corroborate our hypothesis.

5 Conclusion

We proposed a new notion of sparsity for vectors named PQ Index (PQI), which follows the principles a sparsity measure should obey. We develop a new perspective on the compressibility of neural networks by measuring the sparsity of pruned models. We postulate a hypothesis on the relationship between the sparsity and compressibility of neural networks. Motivated by our proposed PQI and hypothesis, we further develop a Sparsity-informed Adaptive Pruning (SAP) algorithm that uses PQI to choose the pruning ratio adaptively. Our experimental results demonstrate that SAP can compress more efficiently and robustly than state-of-the-art algorithms.

REFERENCES

Mehmet Akçakaya and Vahid Tarokh. A frame construction and a universal distortion bound for sparse representations. *IEEE Transactions on Signal Processing*, 56(6):2443–2450, 2008.

Sanjeev Arora, Rong Ge, Behnam Neyshabur, and Yi Zhang. Stronger generalization bounds for deep nets via a compression approach. In *Proc. ICML*, pp. 254–263, 2018.

Andrew R Barron. Universal approximation bounds for superpositions of a sigmoidal function. *IEEE Transactions on Information Theory*, 39(3):930–945, 1993.

Cenk Baykal, Lucas Liebenwein, Igor Gilitschenski, Dan Feldman, and Daniela Rus. Data-dependent coresets for compressing neural networks with applications to generalization bounds. *arXiv preprint arXiv:1804.05345*, 2018.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Proc. NeurIPS*, 33:1877–1901, 2020.

Scott Shaobing Chen, David L Donoho, and Michael A Saunders. Atomic decomposition by basis pursuit. *SIAM review*, 43(1):129–159, 2001.

Hugh Dalton. The measurement of the inequality of incomes. *The Economic Journal*, 30(119): 348–361, 1920.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

David L Donoho. Compressed sensing. *IEEE Transactions on Information Theory*, 52(4):1289–1306, 2006.

Utku Evci, Trevor Gale, Jacob Menick, Pablo Samuel Castro, and Erich Elsen. Rigging the lottery: Making all tickets winners. In *International Conference on Machine Learning*, pp. 2943–2952. PMLR, 2020.

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

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

Corrado Gini. *Variabilità e mutabilità: contributo allo studio delle distribuzioni e delle relazioni statistiche.[Fasc. I.]*. Tipogr. di P. Cuppini, 1912.

Masafumi Hagiwara. Removal of hidden units and weights for back propagation networks. In *Proc. IJCNN*, volume 1, pp. 351–354, 1993.

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

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*, pp. 770–778, 2016a.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Identity mappings in deep residual networks. In *Proc. ECCV*, pp. 630–645. Springer, 2016b.

Yihui He, Xiangyu Zhang, and Jian Sun. Channel pruning for accelerating very deep neural networks. In *Proc. ICCV*, pp. 1389–1397, 2017.

Torsten Hoefler, Dan Alistarh, Tal Ben-Nun, Nikoli Dryden, and Alexandra Peste. Sparsity in deep learning: Pruning and growth for efficient inference and training in neural networks. *J. Mach. Learn. Res.*, 22(241):1–124, 2021.

Hengyuan Hu, Rui Peng, Yu-Wing Tai, and Chi-Keung Tang. Network trimming: A data-driven neuron pruning approach towards efficient deep architectures. *arXiv preprint arXiv:1607.03250*, 2016.

Niall Hurley and Scott Rickard. Comparing measures of sparsity. *IEEE Transactions on Information Theory*, 55(10):4723–4741, 2009.

Lou Jost. Entropy and diversity. *Oikos*, 113(2):363–375, 2006.

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

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Proc. NeurIPS*, 25, 2012.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Communications of the ACM*, 60(6):84–90, 2017.

Yann LeCun, John Denker, and Sara Solla. Optimal brain damage. *Proc. NeurIPS*, 2, 1989.

Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proc. IEEE*, 86(11):2278–2324, 1998.

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

Ilya Loshchilov and Frank Hutter. Sgdr: Stochastic gradient descent with warm restarts. *arXiv preprint arXiv:1608.03983*, 2016.

Jian-Hao Luo, Jianxin Wu, and Weiyao Lin. Thinet: A filter level pruning method for deep neural network compression. In *Proc. ICCV*, pp. 5058–5066, 2017.

Ben Mussay, Margarita Osadchy, Vladimir Braverman, Samson Zhou, and Dan Feldman. Data-independent neural pruning via coresets. *arXiv preprint arXiv:1907.04018*, 2019.

Aaron van den Oord, Sander Dieleman, Heiga Zen, Karen Simonyan, Oriol Vinyals, Alex Graves, Nal Kalchbrenner, Andrew Senior, and Koray Kavukcuoglu. Wavenet: A generative model for raw audio. *arXiv preprint arXiv:1609.03499*, 2016.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. Improving language understanding by generative pre-training. 2018.

Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *Proc. CVPR*, pp. 779–788, 2016.

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

Scott Rickard and Maurice Fallon. The gini index of speech. In *Proceedings of the 38th Conference on Information Science and Systems (CISS'04)*, 2004.

Steffen Schneider, Alexei Baevski, Ronan Collobert, and Michael Auli. wav2vec: Unsupervised pre-training for speech recognition. *arXiv preprint arXiv:1904.05862*, 2019.

Mohammadreza Soltani, Suya Wu, Jie Ding, Robert Ravier, and Vahid Tarokh. On the information of feature maps and pruning of deep neural networks. In *Proc. ICPR*, pp. 6988–6995, 2021.

Robert Tibshirani. Regression shrinkage and selection via the lasso. *J. Royal Stat. Soc. B*, 58(1): 267–288, 1996.

Zhan Wang, Sandra Paterlini, Fuchang Gao, and Yuhong Yang. Adaptive minimax regression estimation over sparse l_q-hulls. *J. Mach. Learn. Res.*, 15(1):1675–1711, 2014.

Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*, 2017.

Wenjing Yang, Ganghua Wang, Jie Ding, and Yuhong Yang. A theoretical understanding of neural network compression from sparse linear approximation. *arXiv preprint arXiv:2206.05604*, 2022.

Mao Ye, Chengyue Gong, Lizhen Nie, Denny Zhou, Adam Klivans, and Qiang Liu. Good subnetworks provably exist: Pruning via greedy forward selection. In *Proc. ICML*, pp. 10820–10830, 2020.

Sergey Zagoruyko and Nikos Komodakis. Wide residual networks. *arXiv preprint arXiv:1605.07146*, 2016.

Appendix

A Limitations and Future Work

Theory We leverage our proposed PQ Index and derive a PQI-bound to indicate the number of retained parameters r. Our result is a lower bound for r, which only suggests the maximum number of model parameters we should prune. One potential future work is to develop an upper for r so that we can better understand the relationship between sparsity and pruning. Furthermore, our result introduces an additional term η_r, which is unavailable before determining the pruning ratio. We treat it as a hyper-parameter in our algorithm and experiments. However, it is desirable to develop a tighter bound without such approximation. Additionally, how p and q together impact the sparsity measure and model pruning has not been thoroughly analyzed. Finally, the theoretical justification of the proposed hypothesis is lacking.

Method Our corroborated hypothesis indicates that a dynamic relationship exists between the model's sparsity and compressibility. However, our proposed SAP algorithm determines the number of pruned parameters based on a static PQI-bound at each iteration. Thus, one potential future work is to further develop the SAP algorithm by considering the dynamics of sparsity. For example, stopping pruning when the PQI starts to increase or the pruning ratio is below some threshold. In this work, we demonstrate the relationship between the performance of iterative pruning and the dynamics of PQI. It is interesting to analyze the critical factors that may determine such dynamics, such as initialization and model architecture. Recent works also introduce gradual magnitude pruning which can outperform iterative pruning algorithms Gale et al. (2019); Renda et al. (2020). Therefore, it is also interesting to study how PQI are related to various pruning methods Evci et al. (2020); Hoefler et al. (2021). We demonstrate that SAP with proper choice of hyper-parameters can outperform LT. It is interesting to explore when SAP can outperform LT with respect to the accuracy-compression trade-off.

Application We apply the proposed PQ Index for model compression because model compression is one of the most important topics related to sparsity. However, many other interesting topics could leverage the PQ Index. For example, one may consider directly optimizing the objective function and PQ Index together for regularization. Furthermore, fairness that advocates similar performances of various groups may also benefit from our PQ Index. Finally, other fields using Gini Index, such as sociology and economics, may also find the proposed alternative index interesting.

B Theoretical Analysis

Proof of Theorem 1:
Note that for any $0 < p < q$, Hölder's inequality gives that

$$\|w\|_q \leq \|w\|_p \leq d^{\frac{1}{p}-\frac{1}{q}}\|w\|_q.$$

We immediately obtain $\mathrm{I}(w) \in [0, 1 - d^{\frac{1}{q}-\frac{1}{p}}]$ from the inequality above.

Next, we prove that I satisfies six properties, which implies that $\mathrm{I}(w)$ is larger if w is sparser. Hurley & Rickard (2009) prove that (P1) and (P2) are automatically satisfied as long as (D1)-(D4) are met. Furthermore, we note that $f(x) = 1 - 1/x$ is monotonous for $x > 0$. Therefore, we only have to prove (D1)-(D4) hold for $S(w) = d^{\frac{1}{p}-\frac{1}{q}}\|w\|_q/\|w\|_p$.

(D2) Scaling is automatically satisfied for $S(w)$ since ℓ_q-norm is homogeneous for any $q > 0$, and (D4) Cloning is clear from the definition of $S(w)$.

For (D1) Robin Hood, we only need to prove that for any $w_i > w_j > 0$, the derivative of $f(t)$ at $t = 0$ is negative, where

$$f(t) = \frac{\{(w_i - t)^q + (w_j + t)^q\}^{1/q}}{\{(w_i - t)^p + (w_j + t)^p\}^{1/p}}.$$

Let $a = w_i - t$ and $b = w_j + t$. Take the derivative with respect to t, we have

$$\begin{aligned} f'(t) &= \frac{(a^q + b^q)^{\frac{1}{q}-1}(-a^{q-1} + b^{q-1})(a^p + b^p)^{1/p}}{(a^p + b^p)^{2/p}} \\ &\quad - \frac{(a^q + b^q)^{1/q}(a^p + b^p)^{\frac{1}{p}-1}(-a^{p-1} + b^{p-1})}{(a^p + b^p)^{2/p}} \\ &= f(t)\left[\frac{-a^{q-1} + b^{q-1}}{a^q + b^q} - \frac{-a^{p-1} + b^{p-1}}{a^p + b^p}\right] \\ &= f(t)\frac{(a^{p-q} - b^{p-q})(a+b)a^{q-1}b^{q-1}}{(a^q + b^q)(a^p + b^p)}. \end{aligned}$$

Note that when $t = 0$, $f(0) > 0$ and $a > b > 0$. Thus, $f'(t) < 0$.

Similarly, for (D3) Rising tide, we need to show that $f'(t)$ is negative, where

$$f(t) = \frac{(\sum_{i=1}^d (w_i + t)^q)^{1/q}}{(\sum_{i=1}^d (w_i + t)^p)^{1/p}}.$$

We can verify that

$$f'(0) = f(0)\left[\frac{\sum_{i=1}^d w_i^{q-1}}{\sum_{i=1}^d w_i^q} - \frac{\sum_{i=1}^d w_i^{p-1}}{\sum_{i=1}^d w_i^p}\right].$$

Thus, we conclude the proof by showing that $h(t) = (\sum_{i=1}^d w_i^{t-1})/(\sum_{i=1}^d w_i^t)$ is a monotonously decreasing function for $t > 0$. This is done by showing $h'(t) < 0$ for all $t > 0$. Actually, since $w_i \geq 0$ and w_i's are not all the same, we know

$$\begin{aligned} h'(t) &= \frac{(\sum_{i=1}^d w_i^{t-1}\ln(w_i))(\sum_{i=1}^d w_i^t) - (\sum_{i=1}^d w_i^{t-1})(\sum_{i=1}^d w_i^t \ln(w_i))}{(\sum_{i=1}^d w_i^t)^2} \\ &= \frac{\sum_{1\leq i<j\leq d}(w_i - w_j)(\ln(w_i) - \ln(w_j))w_i^{t-1}w_j^{t-1}}{(\sum_{i=1}^d w_i^t)^2} < 0. \end{aligned}$$

Proof of Theorem 2:

Recall that M_r is the largest r components of w, and η_r is a constant such that $\sum_{i\notin M_r} |w_i|^p \leq \eta_r \sum_{i\in M_r} |w_i|^p$. Therefore,

$$\begin{aligned} \|w\|_p &= \left(\sum_{1\leq i\leq d} |w_i|^p\right)^{\frac{1}{p}} = \left(\sum_{i\in M_r} |w_i|^p + \sum_{i\notin M_r} |w_i|^p\right)^{\frac{1}{p}} \\ &\leq \left(\sum_{i\in M_r} |w_i|^p + \eta_r \sum_{i\in M_r} |w_i|^p\right)^{\frac{1}{p}} = \left(\sum_{i\in M_r} |w_i|^p\right)^{\frac{1}{p}} (1+\eta_r)^{\frac{1}{p}} \\ &\leq \left(\sum_{i\in M_r} |w_i|^q\right)^{\frac{1}{q}} r^{\frac{1}{p}-\frac{1}{q}}(1+\eta_r)^{\frac{1}{p}} \leq \|w\|_q r^{\frac{1}{p}-\frac{1}{q}}(1+\eta_r)^{\frac{1}{p}}. \end{aligned}$$

Rearranging the above inequality gives

$$r \geq d(1+\eta_r)^{-q/(q-p)}[1 - \mathrm{I}(w)]^{\frac{qp}{q-p}}.$$

C EXPERIMENTAL SETUP

Table 1 and 2 summarizes the model architecture of MLP and CNN used in our experiments. Table 3 shows the statistics of model architecture and hyper-parameters used in our experiments.

Table 1: The model architecture of Multi-Layer Perceptron (MLP) used in our experiments. The n_c, H, W represent the shape of images, namely the number of image channels, height, and width, respectively. K is the number of classes in the classification task. The ReLU layers follow Linear(input channel size, output channel size) layers, apart from the last one.

Image $x \in \mathbb{R}^{n_c \times H \times W}$
Linear($n_c \times H \times W$, 128)
Linear(128, 256)
Linear(256, K)

Table 2: The model architecture of Convolutional Neural Networks (CNN) used in our experiments. The n_c, H, W represent the shape of images, namely the number of image channels, height, and width, respectively. K is the number of classes in the classification task. The BatchNorm and ReLU layers follow Conv2d(input channel size, output channel size, kernel size, stride, padding) layers. The MaxPool2d(output channel size, kernel size) layer reduces the height and width by half.

Image $x \in \mathbb{R}^{n_c \times H \times W}$
Conv2d(n_c, 64, 3, 1, 1)
MaxPool2d(64, 2)
Conv2d(64, 128, 3, 1, 1)
MaxPool2d(128, 2)
Conv2d(128, 256, 3, 1, 1)
MaxPool2d(256, 2)
Conv2d(256, 512, 3, 1, 1)
MaxPool2d(512, 2)
Global Average Pooling
Linear(512, K)

Table 3: Statistics of the models and hyper-parameters used in our experiments for training and pruning.

Dataset		FashionMNIST				CIFAR10			
Model Architecture		Linear	MLP	CNN	ResNet18	Linear	MLP	CNN	ResNet18
Model Size		7.9 K	136.1 K	1.6 M	11.2 M	30.7 K	428.9 K	1.6 M	11.2 M
FLOPS		3.9 M	67.8 M	20.1 G	114.4 G	15.4 M	214.2 M	29.4 G	139.4 G
Train	Epoch E	200							
	Batch size	250							
	Optimizer	SGD							
	Learning rate	1E-01							
	Momentum	0.9							
	Weight decay	5E-04							
	Nesterov	✓							
	Scheduler	Cosine Annealing (Loshchilov & Hutter, 2016)							
Prune	T	30		15		30		15	
	P	0.2							

D EXPERIMENTAL RESULTS

D.1 PQ INDEX

We visualize the PQ Index of pruned models at the global scale with various combinations of p and q. We use zero to indicate the numerical overflow may happen when $p = 0.1$. Note that we use $p = 0.5$ and $q = 1.0$ to compute PQ Index for other figures. The results show that various combinations of p and q also corroborate our hypothesis in different scales, e.g (d) One Shot in Figure 7 and (b) SAP ($p = 1.0$, $q = 2.0$) of Figure 8.

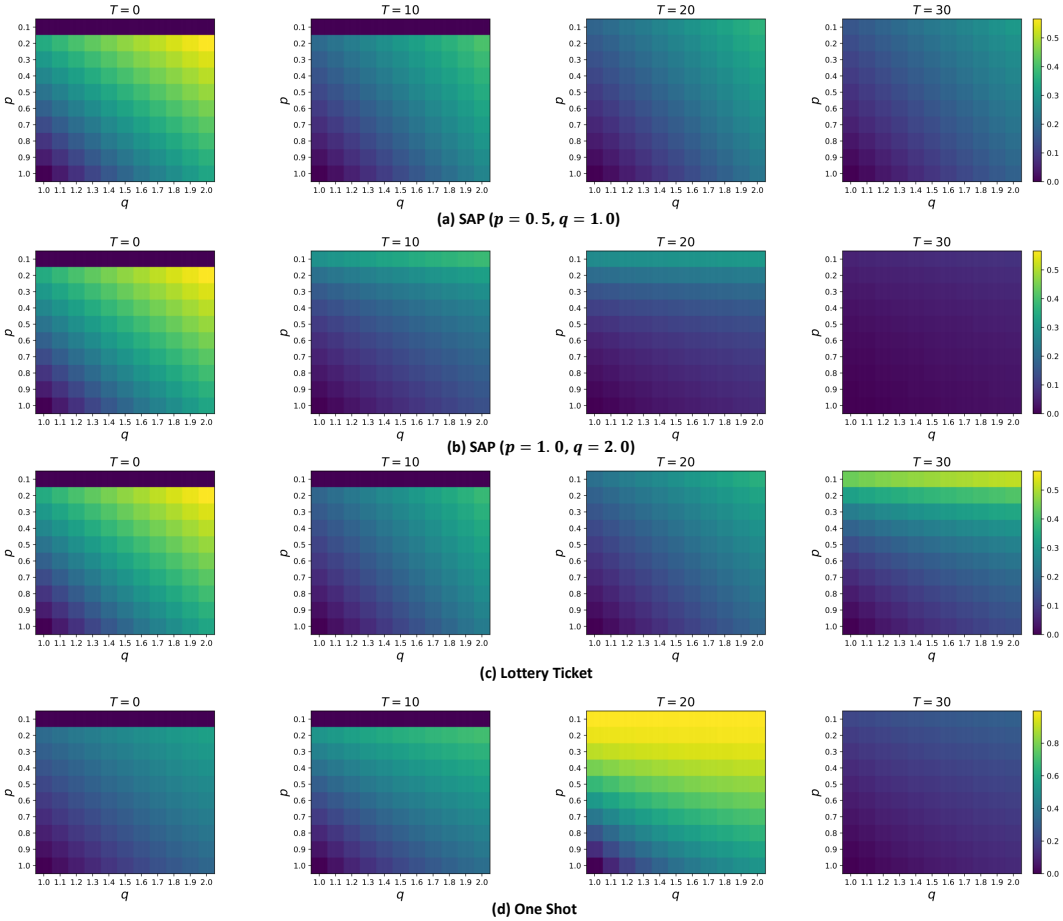

Figure 7: Results of PQ index visualized with various combinations of p and q for FashionMNIST and MLP.

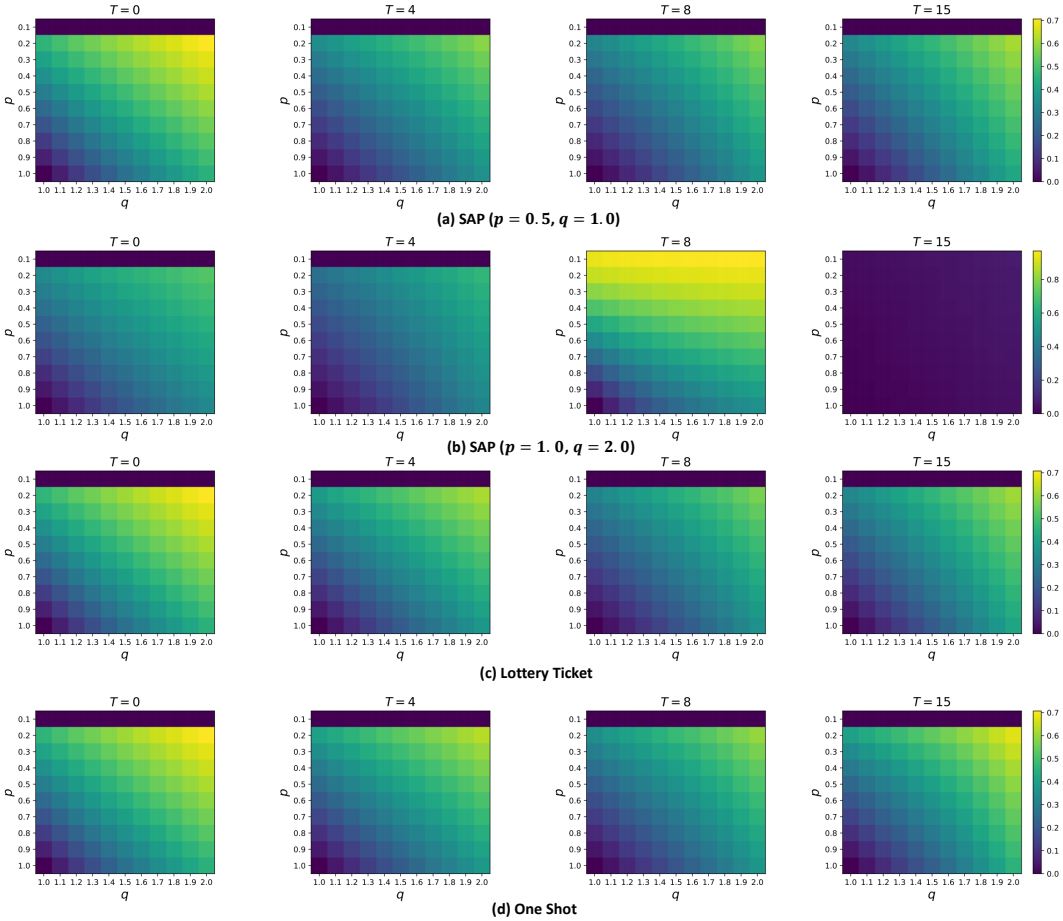

Figure 8: Results of PQ index visualized with various combinations of p and q for CIFAR10 and ResNet18.

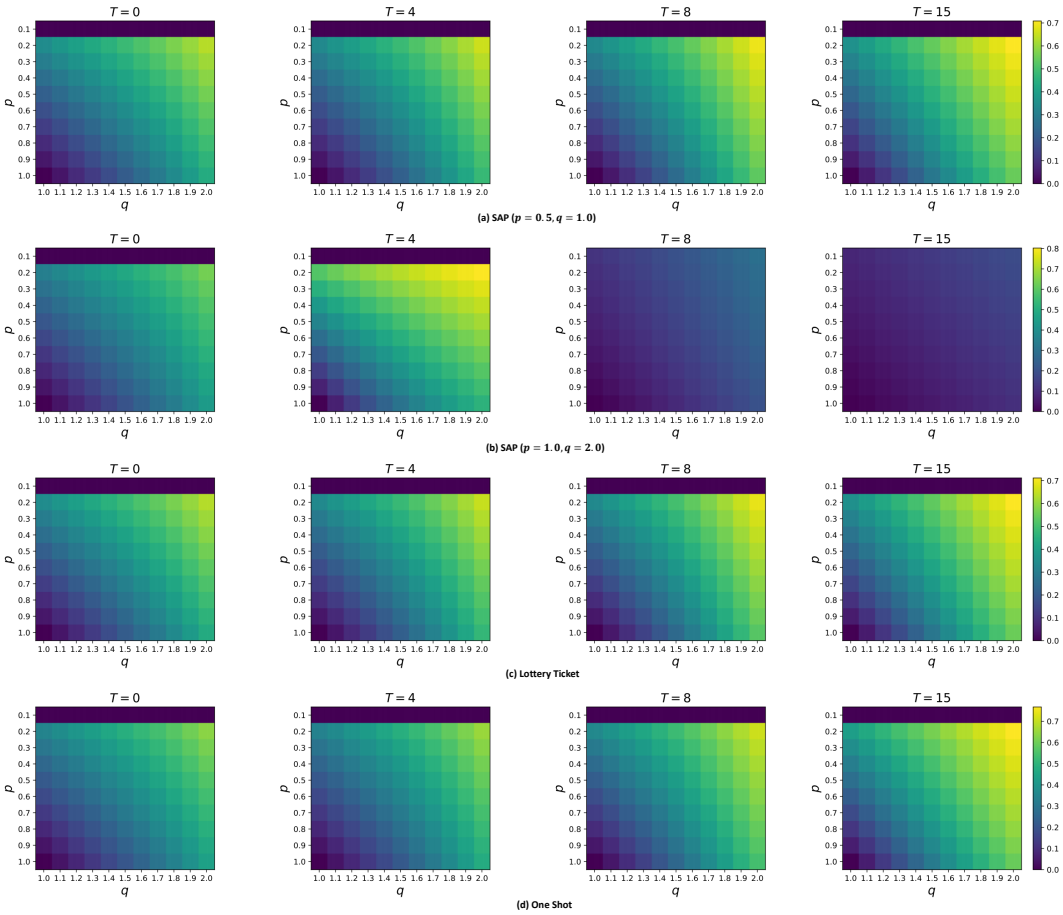

Figure 9: Results of PQ index visualized with various combinations of p and q for CIFAR100 and WResNet28x8.

D.2 RETRAINED AND PRUNED MODELS

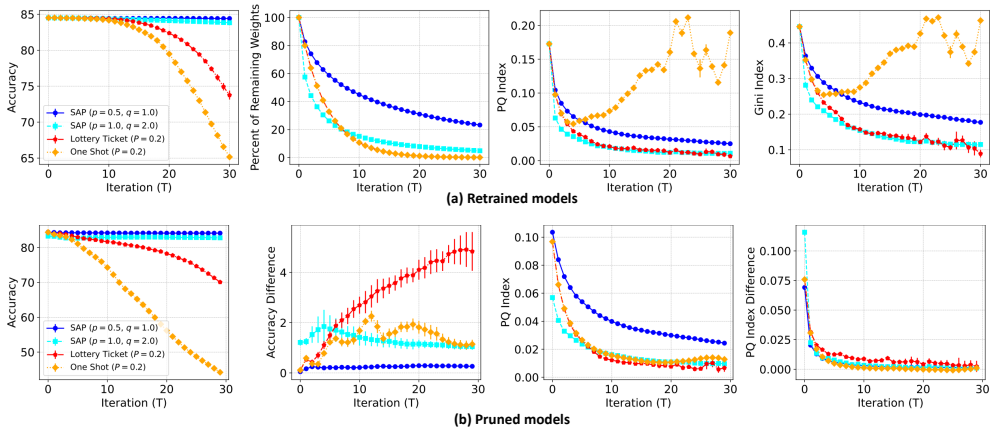

Figure 10: Results of (a) retrained and (b) pruned models at each pruning iteration for 'Global Pruning' with FashionMNIST and Linear.

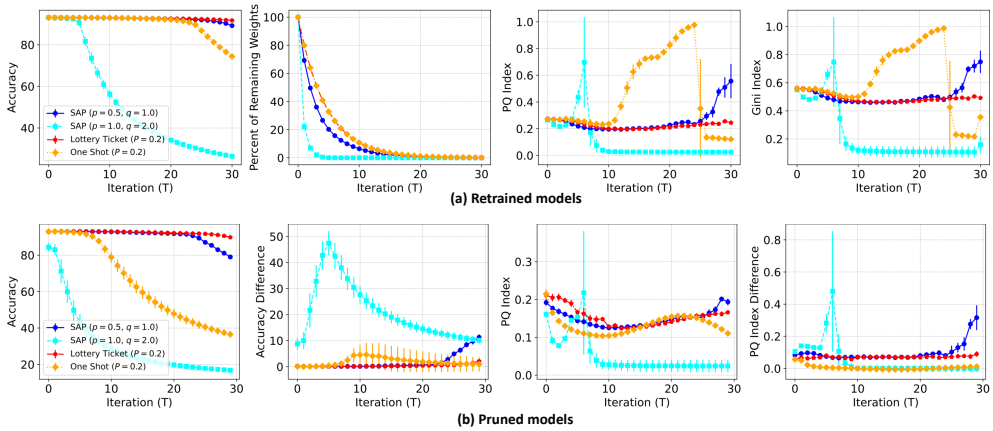

Figure 11: Results of (a) retrained and (b) pruned models at each pruning iteration for 'Global Pruning' with FashionMNIST and CNN.

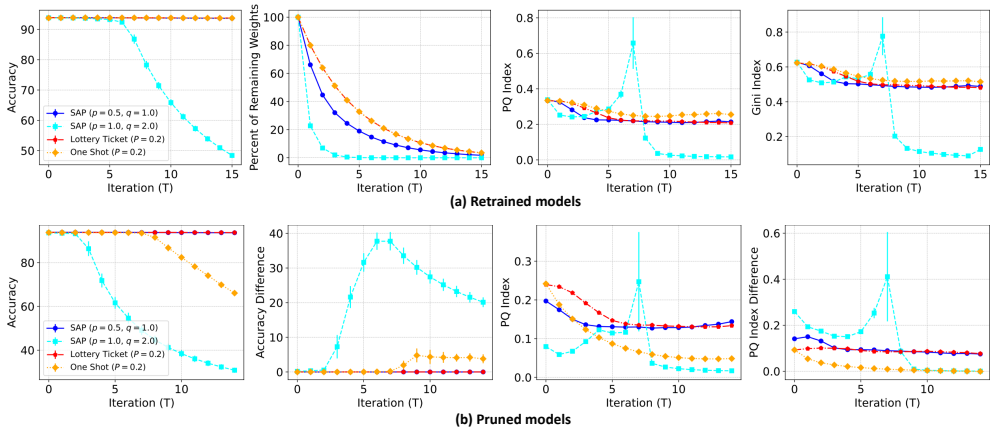

Figure 12: Results of (a) retrained and (b) pruned models at each pruning iteration for 'Global Pruning' with FashionMNIST and ResNet18.

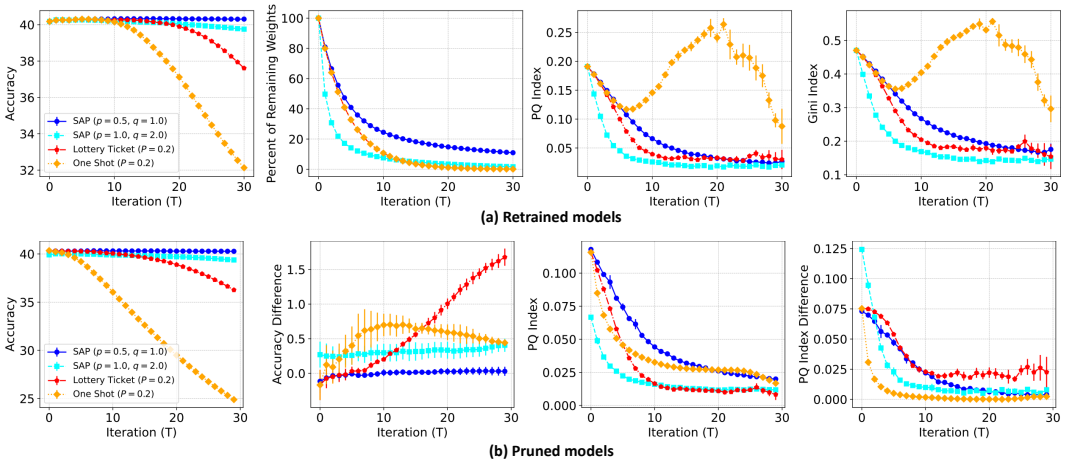

Figure 13: Results of (a) retrained and (b) pruned models at each pruning iteration for 'Global Pruning' with CIFAR10 and Linear.

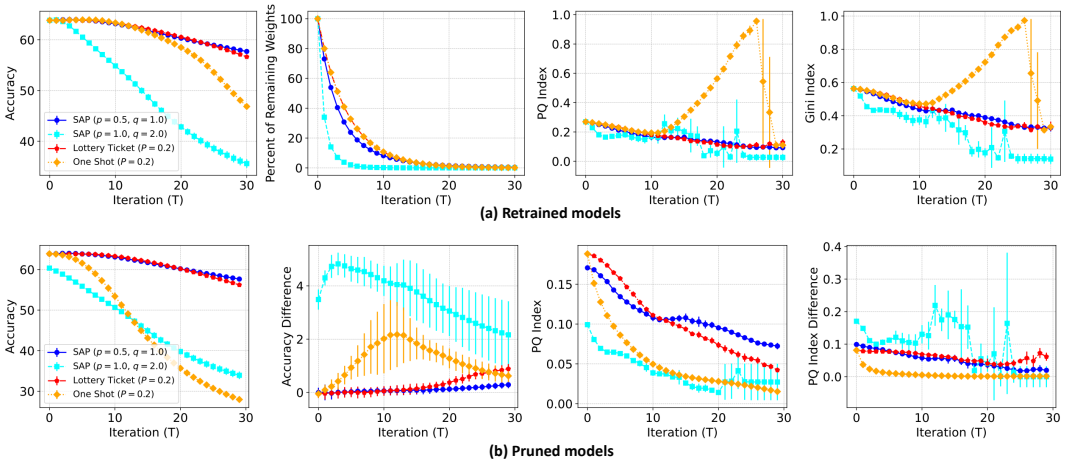

Figure 14: Results of (a) retrained and (b) pruned models at each pruning iteration for 'Global Pruning' with CIFAR10 and MLP.

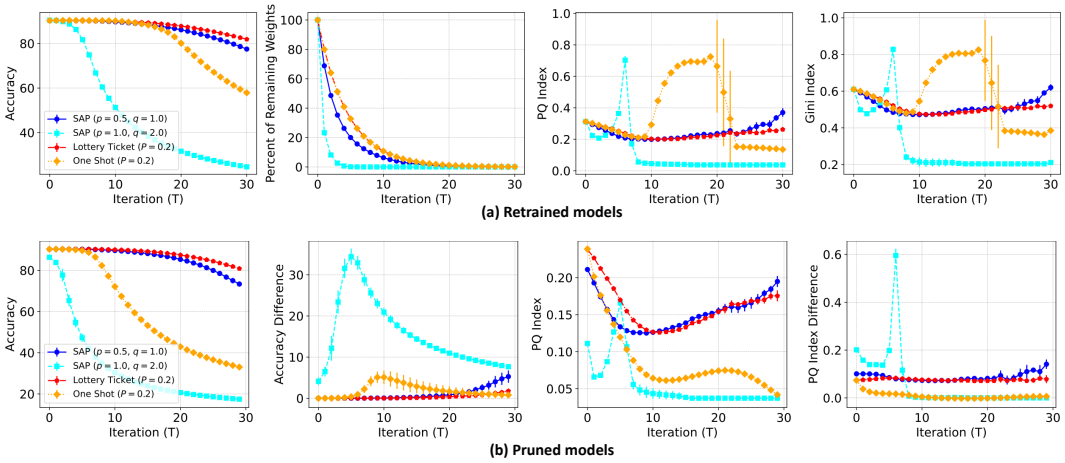

Figure 15: Results of (a) retrained and (b) pruned models at each pruning iteration for 'Global Pruning' with CIFAR10 and CNN.

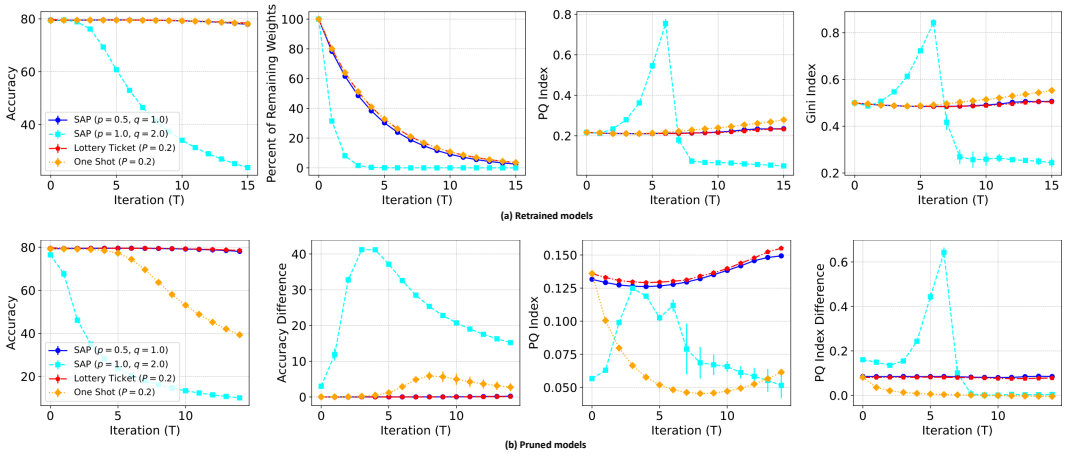

Figure 16: Results of (a) retrained and (b) pruned models at each pruning iteration for 'Global Pruning' with CIFAR100 and WResNet28x8.

D.3 PRUNING SCOPES

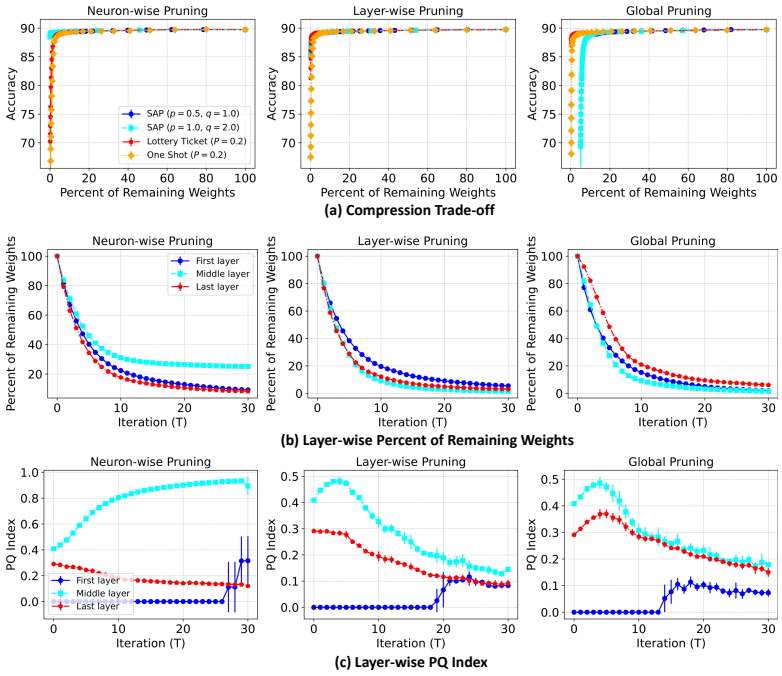

Figure 17: Results of various pruning scopes regarding (a) compression trade-off, (b) layer-wise percent of remaining weights, and (c) layer-wise PQ Index for FashionMNIST and MLP. (b, c) are performed with SAP ($p = 0.5$, $q = 1.0$).

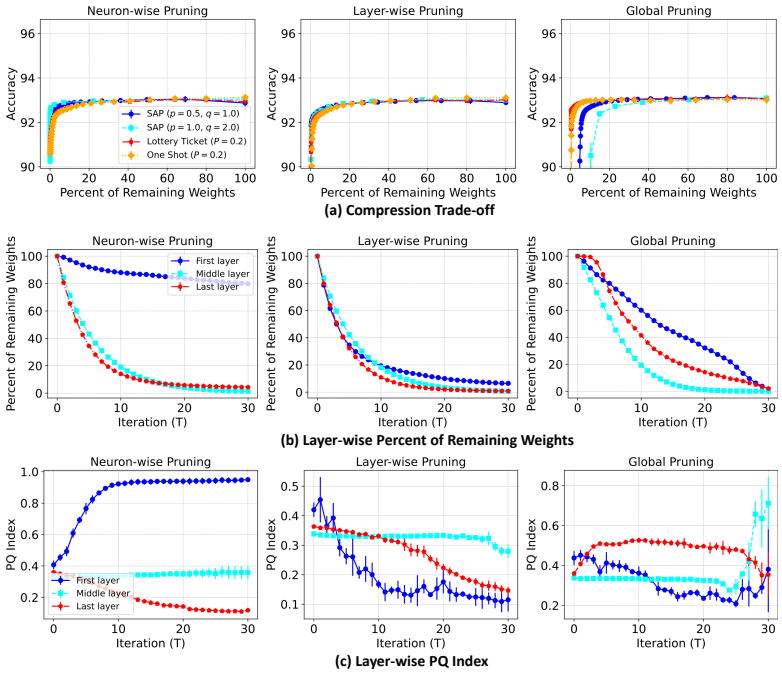

Figure 18: Results of various pruning scopes regarding (a) compression trade-off, (b) layer-wise percent of remaining weights, and (c) layer-wise PQ Index for FashionMNIST and CNN. (b, c) are performed with SAP ($p = 0.5$, $q = 1.0$).

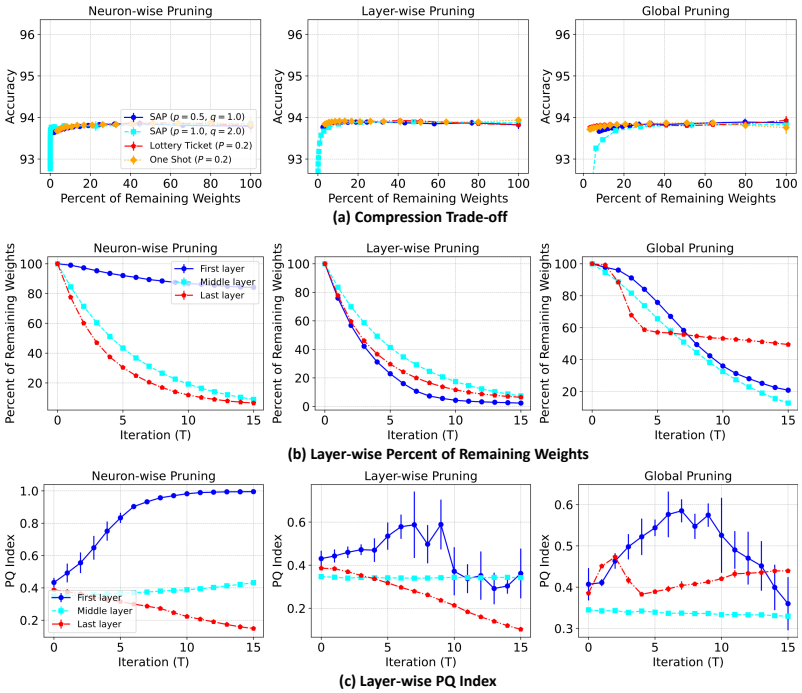

Figure 19: Results of various pruning scopes regarding (a) compression trade-off, (b) layer-wise percent of remaining weights, and (c) layer-wise PQ Index for FashionMNIST and ResNet18. (b, c) are performed with SAP ($p = 0.5$, $q = 1.0$).

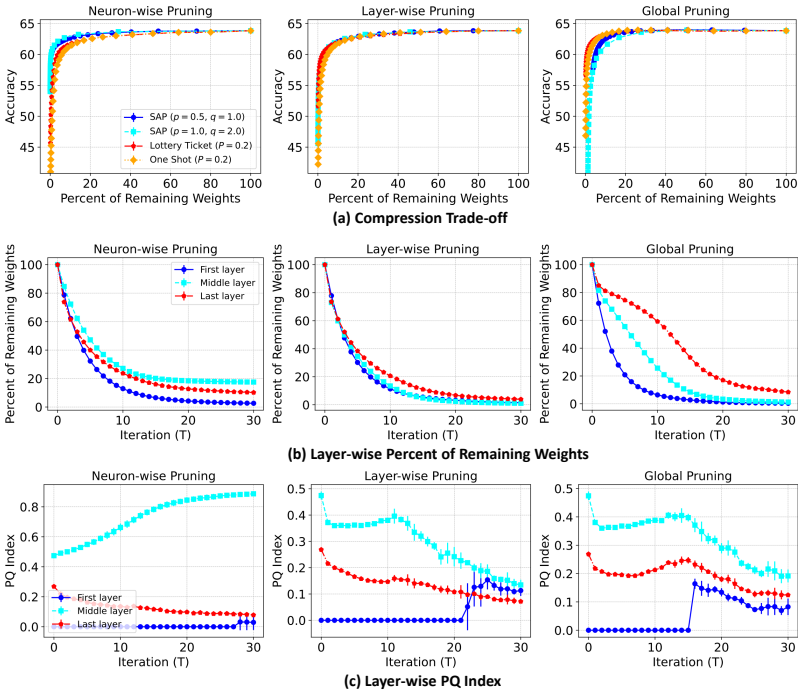

Figure 20: Results of various pruning scopes regarding (a) compression trade-off, (b) layer-wise percent of remaining weights, and (c) layer-wise PQ Index for CIFAR10 and MLP. (b, c) are performed with SAP ($p = 0.5$, $q = 1.0$).

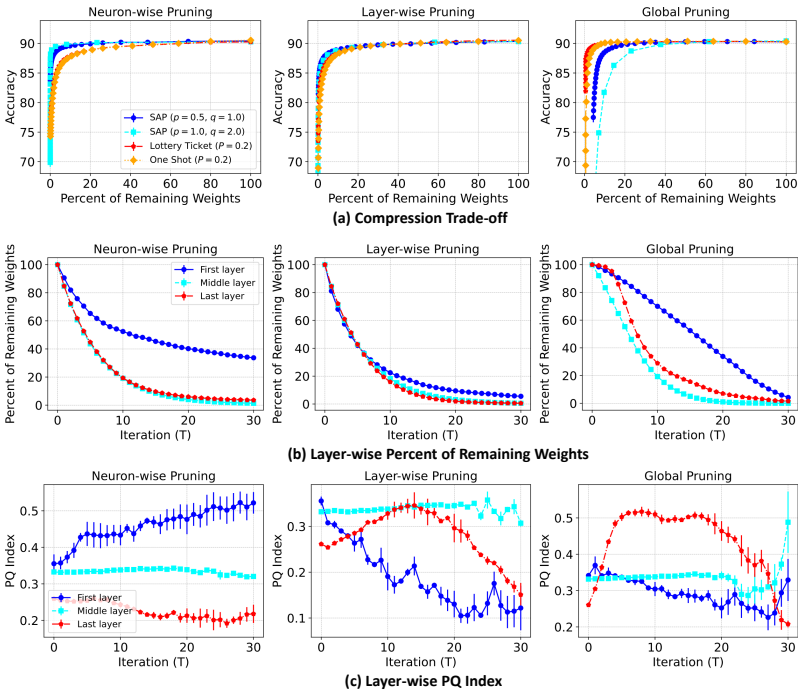

Figure 21: Results of various pruning scopes regarding (a) compression trade-off, (b) layer-wise percent of remaining weights, and (c) layer-wise PQ Index for CIFAR10 and CNN. (b, c) are performed with SAP ($p = 0.5$, $q = 1.0$).

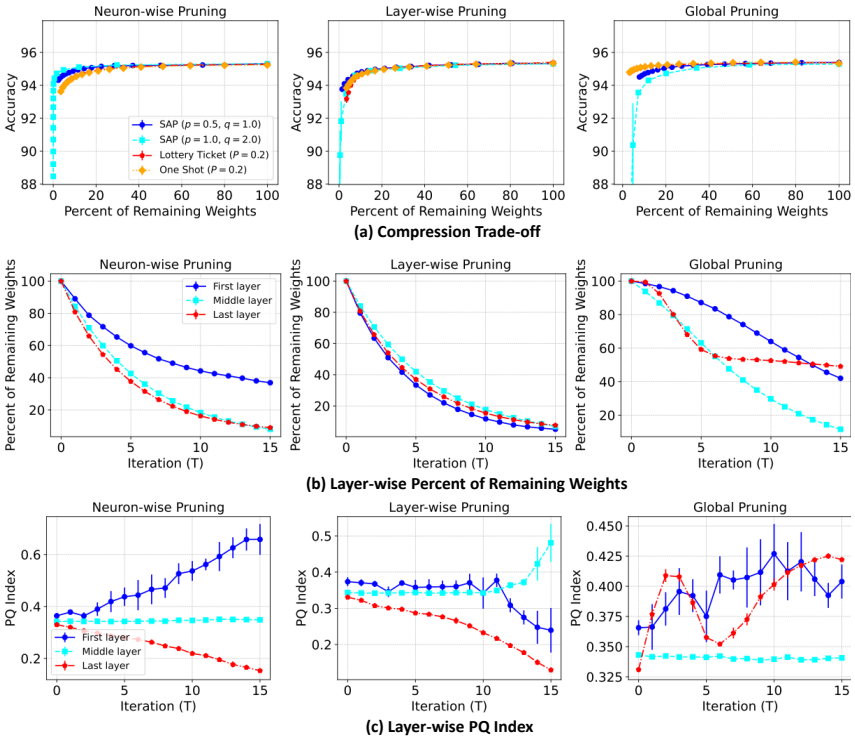

Figure 22: Results of various pruning scopes regarding (a) compression trade-off, (b) layer-wise percent of remaining weights, and (c) layer-wise PQ Index for CIFAR10 and ResNet18. (b, c) are performed with SAP ($p = 0.5$, $q = 1.0$).

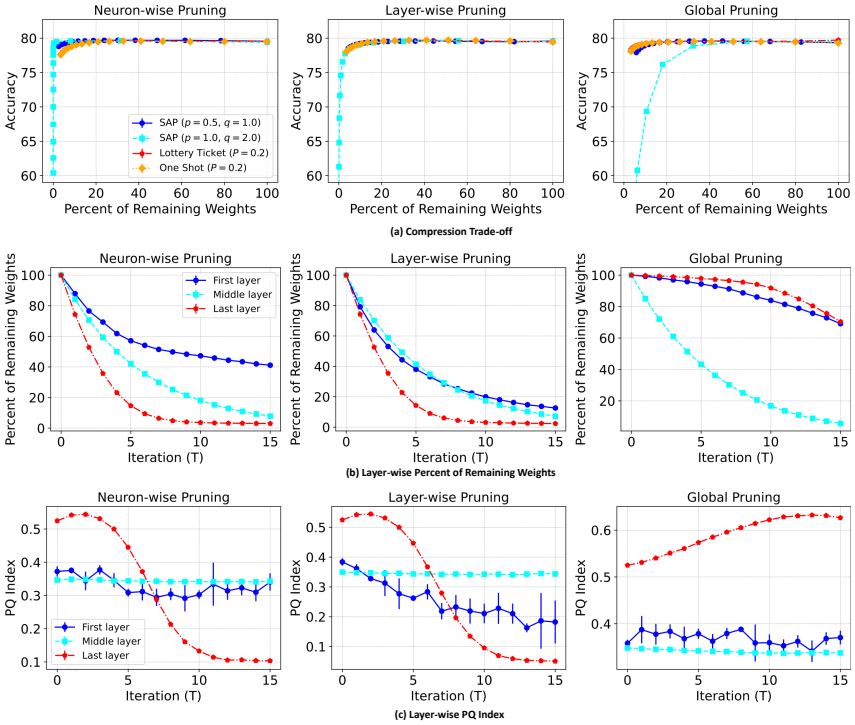

Figure 23: Results of various pruning scopes regarding (a) compression trade-off, (b) layer-wise percent of remaining weights, and (c) layer-wise PQ Index for CIFAR100 and WResNet28x8. (b, c) are performed with SAP ($p = 0.5$, $q = 1.0$).

D.4 EFFECTS OF p AND q

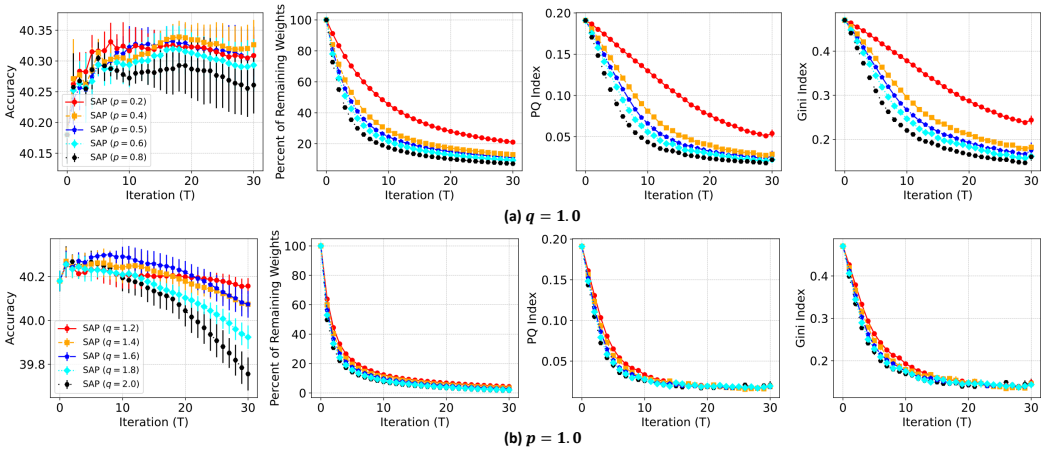

Figure 24: Ablation studies of p and q for global pruning with CIFAR10 and Linear.

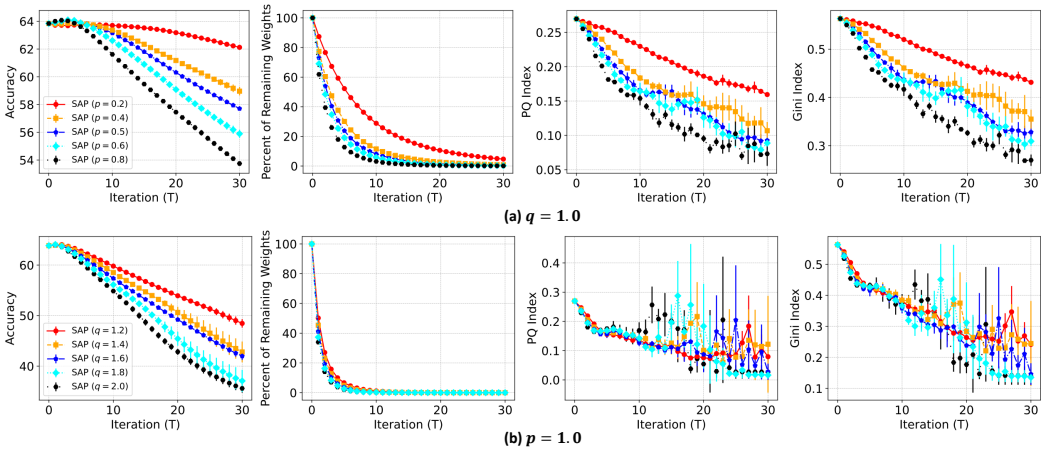

Figure 25: Ablation studies of p and q for global pruning with CIFAR10 and MLP.

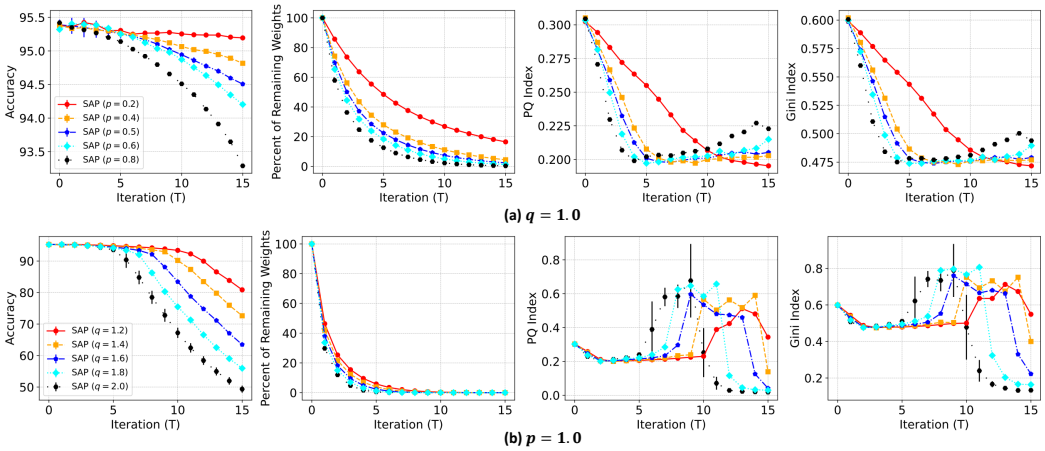

Figure 26: Ablation studies of p and q for global pruning with CIFAR10 and ResNet18.

D.5 EFFECTS OF η_r AND γ

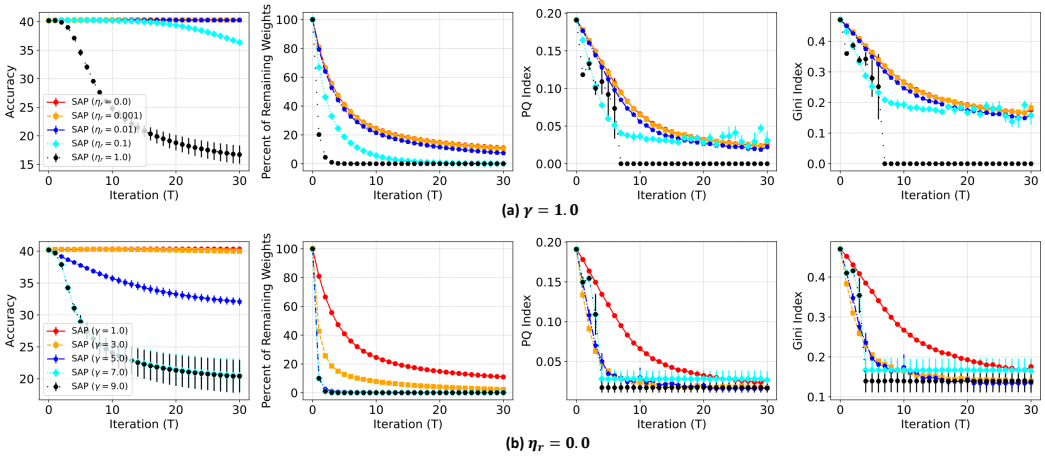

Figure 27: Ablation studies of η_r and γ for global pruning with CIFAR10 and Linear.

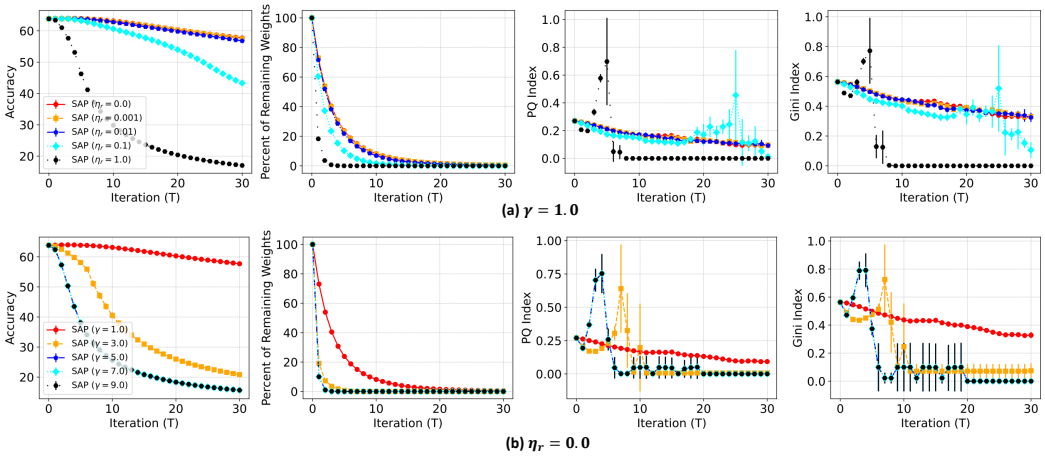

Figure 28: Ablation studies of η_r and γ for global pruning with CIFAR10 and MLP.

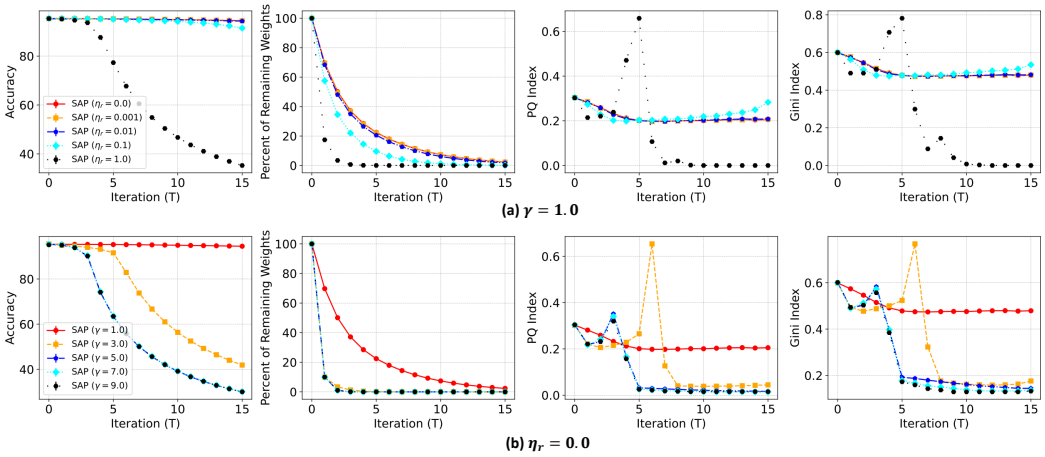

Figure 29: Ablation studies of η_r and γ for global pruning with CIFAR10 and ResNet18.