Noname manuscript No.
(will be inserted by the editor)

Learning Accurate Performance Predictors for Ultrafast Automated Model Compression

Ziwei Wang · Jiwen Lu · Han Xiao · Shengyu Liu · Jie Zhou

Received: date / Accepted: date

Abstract In this paper, we propose an ultrafast automated model compression framework called SeerNet for flexible network deployment. Conventional non-differentiable methods discretely search the desirable compression policy based on the accuracy from exhaustively trained lightweight models, and existing differentiable methods optimize an extremely large supernet to obtain the required compressed model for deployment. They both cause heavy computational cost due to the complex compression policy search and evaluation process. On the contrary, we obtain the optimal efficient networks by directly optimizing the compression policy with an accurate performance predictor, where the ultrafast automated model compression for various computational cost constraint is achieved without complex compression policy search and evaluation. Specifically, we first train the performance predictor based on the accuracy from uncertain compression policies actively selected by efficient evolutionary search, so that informative supervision is provided to learn the accurate performance predictor with acceptable cost. Then we leverage the gradient that maximizes the predicted performance under the barrier complexity constraint for ultrafast acquisition of the desirable compression policy, where adaptive update stepsizes with momentum are employed to enhance optimality of the acquired pruning and quantization strategy. Compared with the state-of-the-art automated model compression methods, experimental results on image classification and object detection show that our method achieves competitive accuracy-complexity trade-offs with significant reduction of the search cost. Code is available at https://github.com/ZiweiWangTHU/SeerNet.

Keywords Automated model compression · Performance predictor · Compression policy optimization · Uncertainty estimation · Evolutionary search

Ziwei Wang[1]
E-mail: wang-zw18@mails.tsinghua.edu.cn
Jiwen Lu[1,✉]
E-mail: lujiwen@tsinghua.edu.cn
Han Xiao[1]
E-mail: h-xiao20@mails.tsinghua.edu.cn
Shengyu Liu[1]
E-mail: liusheng17 @mails.tsinghua.edu.cn
Jie Zhou[1]
E-mail: jzhou@tsinghua.edu.cn

[1] State Key Lab of Intelligent Technologies and Systems, Beijing National Research Center for Information Science and Technology (BNRist), Department of Automation, Tsinghua University, Beijing, 100084, China

1 Introduction

Deep neural networks have achieved the state-of-the-art performance on a wide range of vision tasks such as image classification (He et al., 2016; Phan et al., 2019; Simonyan and Zisserman, 2014), object detection (Liu et al., 2016; Ren et al., 2015), video analysis (Feichtenhofer et al., 2019; Wang et al., 2019b) and many others. Nevertheless, deploying deep neural networks on mobile devices with limited resources for inference is usually impractical due to the heavy computational and storage complexity. Moreover, parameters in well-trained networks are proven to be highly redundant (Denil et al., 2013). Therefore, it is necessary to compress deep neural networks according to hardware configurations for flexible deployment.

In order to reduce the complexity of deep models, network pruning (He et al., 2017; Liu et al., 2018b; Molchanov et al., 2019) and quantization methods (Wang et al., 2021a, 2022a,b) have been widely studied, which

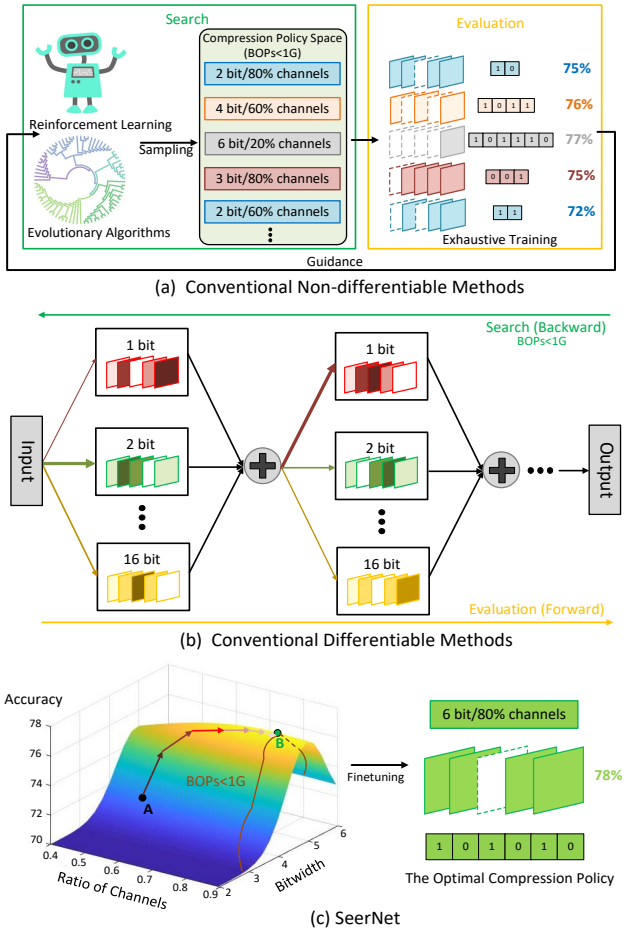


Fig. 1 Comparison between (a) non-differentiable methods, (b) differentiable methods and (c) our SeerNet.Conventional non-differentiable and differentiable frameworks both result in heavy computational cost due to the complex compression policy search and evaluation stages. Our SeerNet directly optimizes the compression policy with a learned accurate performance predictor via efficient gradient ascent, and obtains the networks for deployment by finetuning the lightweight models compressed via selected pruning and quantization policies.

also degrade the model performance due to the network capacity decreases. Pruning removes redundant model components that have little impact on performance, and quantization decreases the bitwidth of weights and activations with low-precision Multiply-Accumulate operations (MACs). Because the hardware resources vary across different deployment scenarios, selecting the optimal compression policy under the device constraint is important to obtain the ideal performance. Hardware equipment with strict resource limit should adopt extremely compressed models to satisfy the complexity constraint, while that with adequate resources only requires slight network complexity reduction to achieve high performance. To accomplish this, automated model compression methods have been proposed, where the optimal pruning ratio or the bitwidth for each convolutional layer is chosen according to the accuracy-complexity trade-off. Non-differentiable methods (He et al., 2018b; Lou et al., 2019; Wang et al., 2019a) applying reinforcement learning and evolutionary algorithms discretely search the optimal compression policy based on the accuracy from exhaustively trained lightweight models, and differentiable approaches (Qu et al., 2020; Wang et al., 2020b, 2021b) optimize the component weights in an extremely large supernet containing all compression policies to acquire the desired lightweight model for deployment. However, the complex compression policy search and evaluation process in both non-differentiable and differentiable methods leads to heavy search cost for automated model compression. For example, the hardware configurations of mobile devices can be selected from different GPUs, FPGAs and many others, and the battery levels can also vary during usage. Therefore, conventional methods with heavy search cost prohibit flexible network deployment due to the frequent changes of model complexity constraint.

In this paper, we present an ultrafast SeerNet framework to learn the optimal model compression policy with the device resource constraint for flexible network deployment. Unlike existing non-differentiable and differentiable methods which undergo the complex compression policy search and evaluation process, our method directly optimizes the compression policy with an accurate performance predictor. The optimal compression policy is obtained via gradient ascent that maximizes the predicted accuracy, so that the efficiency of flexible model deployment is dramatically enhanced via removing resource-exhaustive compression policy search and evaluation. More specifically, we first learn the performance predictor via the accuracy from uncertain compression policies actively selected by evolutionary search, where the uncertainty is estimated via the performance variation with respect to the compression policy perturbation. The actively selected uncertain compression policies offer informative supervision to learn accurate performance predictor with acceptable cost, which can be utilized for flexible model deployment under different hardware scenarios. Then the gradient that maximizes the predicted accuracy under the barrier complexity constraint is leveraged for ultrafast acquisition of the desirable compression policy, and adaptive update step-sizes with momentum are utilized to strengthen the optimality of the obtained pruning and quantization strategy. Figure 1 demonstrates the comparison between our SeerNet and the conventional non-differentiable and differentiable automated model compression methods with complexity constraint calculated by Bit-Operations (BOPs), where our framework achieves ultrafast compression policy selection for flexible network deploy-

ment. Compared with the state-of-the-art automated model compression methods, experiments on the CIFAR-10 (Krizhevsky and Hinton, 2009) and ImageNet (Deng et al., 2009) for image classification and on PASCAL VOC (Everingham et al., 2010) and COCO (Lin et al., 2014) for object detection show that our method achieves competitive performance with significantly reduced search cost. Our contributions are summarized as follows:

(1) We propose the ultrafast compression policy optimization framework which differentiably searches the pruning and quantization strategies on the performance predictor with the highest accuracy under the constraint of computational cost budget.
(2) We present an active compression policy evaluation method that samples the most uncertain pruning and quantization strategies, so that the accurate performance predictor is learned in acceptable training cost with informative supervision.
(3) We conduct extensive experiments on image classification and object detection, and the results consistently show that the presented SeerNet achieves competitive accuracy-complexity trade-offs with significant reduction of compression policy search cost.

2 Related Work

We briefly review three related topics including (1) model compression, (2) AutoML and (3) active learning.

2.1 Model Compression

Pruning and Quantization are two widely adopted strategies for model compression. Pruning aims to remove the unimportant network components that have least influence on the performance, while quantization decreases the bitwidths of network weights and activations and substitutes float MACs with the low-precision ones.

Network pruning has been comprehensively studied in recent years because the model performance is nearly unaffected with sizable complexity degradation. Early attempts (Han et al., 2015a; Liu et al., 2015) cut off the redundant fine-grained neurons and connections in an unstructured manner, which limited the actual acceleration on hardware equipment due to the irregular weight parameters. To address this, channel-pruning methods were later proposed, where the entire convolution channels were pruned according to the defined importance score. He et al. (2017) iteratively selected the channels for pruning with Lasso regression and finetuned the lightweight networks. The definition of the channel importance score have been also widely studied for effective pruning. The L1 and L2 norm of activations were used as the importance score in (Li et al., 2016) and (He et al., 2018a) respectively. Molchanov et al. (2016) and Peng et al. (2019) leveraged the first-order and second-order Taylor expansion with respect to the objective to evaluate the channel importance. Meanwhile, advanced sparsity regularization strategies (Li et al., 2019a, 2020a; Louizos et al., 2017) have been presented to achieve better trade-offs between the model accuracy and complexity. Nevertheless, the uniform pruning ratio across layers for various hardware configurations prohibits flexible network deployment due to the mismatch between hardware resources and model complexity.

Network quantization has been widely adopted in computer vision due to its efficiency in computation and storage, which is divided into one-bit and multi-bit quantization according to the bitwidth of network weights and activations. For the former, Hubara et al. (2016) and Rastegari et al. (2016) binarized weights and activations for efficient inference. Liu et al. (2018a) added an extra shortcut in consecutive layers to enhance the representational capacity of binary neural networks. Gong et al. (2019) optimized the soft quantization strategy so that the discrepancy between the learning objective and the surrogate loss could be minimized. Bethge et al. (2020) increased the quality and capacity of features by channel enlargement and feature refinement, and created the efficient stem architectures to further reduce the computational cost of full-precision layers. Therefore, they even achieved higher accuracies than MobileNetV1 (Howard et al., 2017) with similar computational complexity. Binary neural networks suffer from the extremely low network capacity, and multi-bit networks have been proposed with wider bitwidth and more sufficient representational power. Choi et al. (2018) adaptively selected the activation clipping threshold to learn networks in 2-5 bits with high performance. Zhang et al. (2018) minimized the quantization errors for all weights and activations to alleviate the information loss. Li et al. (2019b) overcome the training instabilities of four-bit object detectors with hardware-friendly implementations. Similar to pruning with uniform compression ratio, fixed-bit quantization cannot satisfy the demand of different deployment scenarios, where platforms with strict resource constraint require highly compressed models and vice versa.

2.2 AutoML

Since the hardware configurations and battery levels vary significantly in different deployment scenarios, ex-

ploiting AutoML for automatic model compression aroused extensive interest in computer vision. The goal of AutoML is to select the compression policy that results in the best performance with the hardware resource constraint. Conventional AutoML frameworks for automatic model compression can be categorized into non-differentiable and differentiable methods based on the search strategy. For the former, He et al. (2018b) and Wang et al. (2019a) applied the reinforcement learning to search the optimal layer-wise pruning and quantization policy respectively according to the accuracy from exhaustively trained networks. Wang et al. (2020a) used the evolutionary algorithms to acquire the desired compression policy and the architectures for the subnets of the once-for-all networks (Cai et al., 2019). For the latter, differentiable methods were presented to deal with the difficulties in discrete optimization of non-differentiable methods. Cai and Vasconcelos (2020) designed an extremely large supernet containing all quantization policies, and adjusted the importance of each quantization policy via back-propagation. Wang et al. (2020b) jointly searched the pruning and quantization policy via variational information bottleneck and the learned quantization mapping. Yu et al. (2020) constructed barrier penalty to ensure the obtained quantization policy satisfying the complexity constraint. However, complex model search and evaluation process in both non-differentiable and differentiable methods causes heavy computational cost for optimal compression policy acquisition, which prohibits the flexible network deployment for various hardware configurations and battery levels. Jin et al. (2020) and Bulat and Tzimiropoulos (2021) trained a once-for-all network that could be quantized to any bits at runtime without finetuning. The once-for-all network quantization methods are orthogonal to automated model compression and can be combined with AutoML for further performance improvement.

2.3 Active Learning

Active learning enforces the model to acquire promising performance with few annotated training samples, where part of the training data providing effective supervision is labeled. The widely adopted criteria for annotation in active learning is based on the informativeness of the selected sample, which is evaluated by the prediction uncertainty. The uncertainty can be defined as the entropy of the posterior distribution (Joshi et al., 2009; Luo et al., 2013; Settles and Craven, 2008), disagreement among different classifiers (Melville and Mooney, 2004; Vasisht et al., 2014; Wu et al., 2022), difference between the largest and the second largest posterior probabilities (Balcan et al., 2007) and the distance to the boundary (Abbasnejad et al., 2020; Li and Guo, 2014; Vijayanarasimhan and Grauman, 2014). Gal et al. (2017) employed deep neural networks to estimate task uncertainty through multiple forward passes in a data-driven manner. Beluch et al. (2018) presented a classifier committee to acquire accurate uncertainty estimation according to the disagreement. Wang et al. (2020c) selected informative samples for hash code learning by considering the pairwise similarity uncertainty. Abbasnejad et al. (2020) generated the most uncertain counterfactual sample with true labels by analyzing the performance sensitivity to the input perturbation. Siddiqui et al. (2020) measured the uncertainty of the semantic segmentation model via the inconsistency in predictions across viewpoints, which significantly lowered the cost of pixel-wise annotation. In this paper, we extend the active learning to efficiently train the accurate performance predictor with acceptable training cost, where only the most uncertain compression policy providing informative supervision is evaluated for actual accuracy acquisition.

3 Approach

In this section, we briefly review automatic model compression, which suffers from the heavy computational cost in compression policy search and evaluation. Then we introduce the details of compression policy optimization via the performance predictor. Finally, we propose active compression policy evaluation to learn the accurate performance predictor.

3.1 Automated Model Compression

The automated model compression is critical for deploying deep neural networks on different portable devices, as it provides the optimal compression policy with different computational cost constraint. The objective of automated model compression is written as follows:

$$\begin{aligned} &\max_{\boldsymbol{\mathcal{S}},\boldsymbol{\theta}} ACC_{val}(\boldsymbol{\mathcal{S}}(\boldsymbol{\mathcal{N}}),\boldsymbol{\theta}) \\ &s.t. \quad C(\boldsymbol{\mathcal{S}}(\boldsymbol{\mathcal{N}})) \leqslant C_0 \end{aligned} \tag{1}$$

where $\boldsymbol{\mathcal{N}}$ and $\boldsymbol{\mathcal{S}}$ are the original networks and the compression policy respectively. $\boldsymbol{\theta}$ represents the parameters of the compressed networks, and ACC_{val} means the accuracy on the validation dataset. $C(\boldsymbol{\mathcal{S}}(\boldsymbol{\mathcal{N}}))$ stands for the complexity of the compressed networks and C_0 is the complexity constraint from device resources.

As shown in Figure 1(a), the non-differentiable methods take turns to search better compression policies and

evaluate the sampled lightweight models. In the evaluation process, all sampled lightweight models are trained exhaustively to obtain the actual performance. During the search stage, agents in reinforcement learning or population in evolutionary algorithms are optimized to achieve higher accuracy with lower complexity, where the updated agents or population sample the best candidates for evaluation. As demonstrated in Figure 1(b), the differentiable methods optimize an extremely large supernet, where different compression policies form parallel modules for each layer. The output of all modules in each layer is added with different importance weights before being fed forward to the next layer. For the evaluation stage, the images are fed forward into the supernet to acquire loss value. For the search stage, importance weights of different modules are updated via back-propagation. The optimal compression policy is obtained by discretizing the soft module weights for the converged supernet.

However, compression policy search and evaluation in both non-differentiable and differentiable methods cause heavy computational cost. The mobile devices can be equipped with various hardware such as different GPUs, FPGAs and many others, and the battery levels can also vary during the usage. Hence, the heavy search cost prohibits flexible network deployment because of the frequent changes of model complexity constraint. Our goal is to remove the resource-exhaustive compression policy search and evaluation to achieve ultrafast automated model compression.

3.2 Ultrafast Compression Policy Optimization

In order to enhance the efficiency of automated model compression, we directly optimize the compression policy according to the learned performance predictor. In this section, we first introduce the learning objectives of compression policy optimization and then detail the compression policy update during the optimization.

3.2.1 Learning Objectives

The performance predictor consists of multi-layer perceptron (MLP), which takes the compression policies across all layers in the backbone architectures as input and predicts the accuracy of the lightweight models. Since the goal of automated model compression is to select the compression policy that leads to the highest accuracy with the given computational cost constraint, the objective J for compression policy optimization is written in the following form:

$$\begin{aligned} &\max J = f(\boldsymbol{s}) \\ &s.t. \quad C(\boldsymbol{\mathcal{N}_s}) \leqslant C_0 \end{aligned} \tag{2}$$

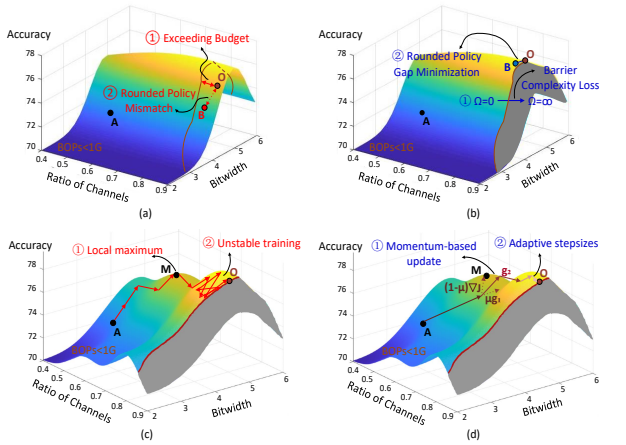


Fig. 2 (a) and (b) visualize the learning objectives of (4) and (5) respectively, where A, O, B represent the initialized policy, the optimal policy after optimization and the discrete policy for deployment. Directly learning (4) fails to rigidly limit the computational complexity of the lightweight model compressed by the searched policy within the budget due to the soft constraint, and results in sizable mismatch between the optimal policy and the rounded one for deployment because of the discrete nature of quantization and pruning policies. In order to address these problems, we present barrier complexity constraint and rounded policy gap minimization in the policy optimization objective. (c) and (d) demonstrate the optimization process of vanilla gradient ascent and (8), where M means the local maximum. Vanilla gradient ascent faces the challenges of local maximum due to the non-convexity and unstable training process because of the fixed stepsizes. On the contrary, we present the momentum-based policy update to escape from the local maximum and propose adaptive stepsizes to stabilize the optimal policy search.

where $f(\boldsymbol{s})$ means the predicted accuracy of the lightweight model with the compression policy $\boldsymbol{s}$. The definition of compression policy is $\boldsymbol{s} = [s_p^1, s_w^1, s_a^1, ..., s_p^L, s_w^L, s_a^L]$, where s_p^i, s_w^i and s_a^i stand for the channel pruning ratio, weight bitwidth and activation bitwidth of the i_{th} layer out of L layers. In our implementation, weight bitwidth s_w^i and activation bitwidth s_a^i are scaled to $\frac{s_w^i}{s_{w,max}}$ and $\frac{s_a^i}{s_{a,max}}$, where $s_{w,max}$ and $s_{a,max}$ respectively represent the largest weight and activation bitwidth in the search space of compression policy. $\boldsymbol{\mathcal{N}_s}$ means the lightweight models obtained by compressing the original networks $\boldsymbol{\mathcal{N}}$ with the compression strategy $\boldsymbol{s}$. The network complexity is defined as Bit-Operations (BOPs) (Bethge et al., 2020; Louizos et al., 2018; Wang et al., 2020b) calculated in the following:

$$C(\boldsymbol{\mathcal{N}_s}) = \sum_{i=1}^{L} s_w^i s_a^i (1 - s_p^{i-1})(1 - s_p^i) \cdot C_{ori}^i \tag{3}$$

where $C_{ori}^i = h_i w_i k_h^i k_w^i c_{i-1} c_i b_w^i b_a^i$ demonstrates the BOPs of the i_{th} layer in the original networks. h_i, w_i and c_i respectively represent the height, width and the number of channels of the output feature map in the

i_{th} layer, and k_h^i and k_w^i stand for the kernel height and width in the i_{th} convolutional layer. For the full-precision networks, the weight and activation bitwidths of the i_{th} layer denoted as b_w^i and b_a^i are usually set as 32. Since BOPs reveal the effect of model complexity decrease induced by network pruning and quantization, we utilize the reduction ratio of BOPs to reflect the compression ratio.

Because better performance is usually obtained by networks with higher capacity, the optimal compression policy for (2) can be obtained when the model complexity achieves the computational cost constraint C_0. In order to efficiently optimize the desirable compression policy, the Lagrange multipliers can be employed to form the surrogate objective function with the hyperparameter λ, which is shown in the following:

$$\max J = f(\boldsymbol{s}) - \lambda C(\boldsymbol{\mathcal{N}_s}) \tag{4}$$

When the optimization completes, rounding the policy to the nearest one on grids yields the pruning and quantization policy for deployment due to their discrete nature. Nevertheless, directly optimizing (4) deviates the obtained compression policy from the optimal one due to the following two reasons. First, the soft model complexity constraint in the objective cannot strictly limit the computational cost of the lightweight networks within the budget, which usually leads to suboptimal policies and huge search cost due to the repeated trials. Second, because the final compression policy for deployment is acquired by rounding the optimal policy to the nearest one on grids, the mismatch between the searched optimal policy and the discrete policy for deployment decreases the accuracy of the lightweight models. In order to address these problems, we formulate the objective function J^* for compression policy optimization containing the barrier complexity constraint and rounded policy gap minimization:

$$\max J^* = f(\boldsymbol{s}) - \lambda_1 \Omega(C(\boldsymbol{\mathcal{N}_s})) - \lambda_2 d(\boldsymbol{s}, \boldsymbol{s}_0) \tag{5}$$

where λ_1 and λ_2 are hyperparameters that demonstrate the importance of different objective terms. $\Omega(C(\boldsymbol{\mathcal{N}_s}))$ means the barrier complexity loss for the lightweight networks $\boldsymbol{\mathcal{N}_s}$, which is assigned to zero for $C(\boldsymbol{\mathcal{N}_s})$ less than C_0 and set to infinity otherwise. $d(\boldsymbol{s}, \boldsymbol{s}_0)$ represents the distance between the compression policy $\boldsymbol{s}$ and its discrete counterpart $\boldsymbol{s}_0$. Figure 2 (a) and (b) visualize the learning objectives of (4) and (5) respectively. The barrier complexity loss in our SeerNet ensures the obtained optimal compression policy to satisfy the computational budget with full utilization of computational resources, and the rounded policy gap is minimized to decrease the performance drop for policy discretization.

Algorithm 1 Compression policy optimization

Input: Network architecture $\mathcal{N}$, performance predictor f, computational cost constraint C_0, compression policy update round Max_iter.
Output: The optimal compression policy $\boldsymbol{s}^*$.
Initialize: Randomly assign $\boldsymbol{s}$ where $C(\mathcal{N}_{\boldsymbol{s}}) \approx \frac{C_0}{2}$.
for $t = 1, 2, ..., Max\_iter$ **do**
  Calculate the objective function of $\boldsymbol{s}_t$ via (5).
  Compute the gradient with respect to $\boldsymbol{s}_t$ by (9).
  Update the compression policy with the above gradient according to (8).
  if $C(\mathcal{N}_{\boldsymbol{s}_t}) \geqslant C_0$ **then return** compression policy $\boldsymbol{s}_t$.
  end if
end for
return compression policy $\boldsymbol{s}_{t+1}$.

In order to enable the barrier complexity loss to be differentiable, we design $\Omega(C(\boldsymbol{\mathcal{N}_s}))$ with the following log-like function adopted from (Finlay et al., 2019; Yu et al., 2020):

$$\Omega(C(\boldsymbol{\mathcal{N}_s})) = -\log(C_0 - C(\boldsymbol{\mathcal{N}_s})) \tag{6}$$

Since $\Omega(C(\boldsymbol{\mathcal{N}_s}))$ is only rapidly amplified by the logarithm when approaching the complexity constraint C_0, the obtained lightweight model is strictly limited by the complexity constraint with full utilization of the computational resources. For the distance between the compression policy $\boldsymbol{s}$ and its discrete counterpart $\boldsymbol{s}_0$, we present the L_2 norm to measure their similarity, which is written as:

$$d(\boldsymbol{s}, \boldsymbol{s}_0) = ||\boldsymbol{s} - \boldsymbol{s}_0||_2^2 \tag{7}$$

where $||\cdot||_2$ represents the L_2 norm. Because $\boldsymbol{s}_0$ is obtained by rounding $\boldsymbol{s}$ to its nearest policy on grids, we relax $\boldsymbol{s}_0$ as a constant (Erin Liong et al., 2015) for gradient back-propagation.

3.2.2 Compression Policy Update

As the predicted accuracy and the complexity of compressed models can both be obtained by the differentiable calculation, we leverage the gradient that maximizes the objective (5) with momentum to update the compression policy:

$$\boldsymbol{s}_{t+1} = \boldsymbol{s}_t + \epsilon_t \cdot \frac{\boldsymbol{g}_t}{||\boldsymbol{g}_t||_2} \tag{8}$$

where $\boldsymbol{s}_t$ means the compression policy in the t_{th} step during the optimization. $\boldsymbol{g}_t$ illustrates the accumulated gradient in the t_{th} step, and ϵ_t is defined as the step-size in the t_{th} step which is adaptively assigned. As indicated in (Dong et al., 2018) that integrating the momentum into iterative processes of input update can

boost optimization, we adopt the accumulated gradients in the following that escape from the local maximum (Duch and Korczak, 1998; Sutskever et al., 2013):

$$\boldsymbol{g}_{t+1} = \mu \cdot \boldsymbol{g}_t + (1-\mu) \cdot \frac{\nabla_{\boldsymbol{s}} J^*}{||\nabla_{\boldsymbol{s}} J^*||_2} \tag{9}$$

where μ is a hyperparameter that balances the momentum and the current gradient in the accumulated gradients. In order to stabilize the training process (Kingma and Ba, 2014), the stepsize for compression policy update in each step should be adjusted with respect to the complexity difference between the current lightweight model and the computational complexity constraint. When the current policy is far from the complexity constraint, the stepsize should be large in order to accelerate training. On the contrary, the stepsize should be small for policy optimization near the computational cost budget due to the extremely large barrier complexity loss, so that fine-grained optimization is adopted to stably search the optimal policy within the complexity constraint. We present the adaptive stepsize at the t_{th} step as follows:

$$\epsilon_t = \eta \cdot (C_0 - C(\boldsymbol{\mathcal{N}}_{\boldsymbol{s}_t})) \tag{10}$$

where η is a hyperparameter and $C(\boldsymbol{\mathcal{N}}_{\boldsymbol{s}_t})$ demonstrates the complexity of the lightweight models compressed by the policy in the t_{th} iterative update step. Figure 2 (c) and (d) illustrate the vanilla gradient ascent and the presented compression policy optimization respectively, where our optimization process escapes from the local maximum and stably obtains the policy with the highest accuracy within the complexity constraint.

The compression policy update process stops until reaching the computational cost constraint or achieving the maximum iteration steps. The detailed procedures of ultrafast compression policy optimization are shown in Algorithm 1, where flexible deployment across different hardware configurations and battery levels is achieved since the gradient of the performance predictor consisting of several MLPs is calculated with extremely little computational cost.

3.3 Learning Performance Predictor via Active Compression Policy Evaluation

The acquisition of the optimal lightweight model via differentiable compression policy optimization requires the learned performance predictor to be precise, where the gap between the predicted and actual performance is negligible. Conventional accuracy predictors for network architecture search (Dai et al., 2019; Wen et al., 2020) randomly sample compression policies, and acquire the actual performance by exhaustively training the lightweight models. Then the actual accuracy is employed to supervise the performance predictor that regresses the accuracy of the compression policy. However, the number of sampled lightweight models for evaluation is extremely small compared with the large space of compression policies due to the limited computational resources. Randomly sampled compression policies fail to provide informative supervision for performance predictor learning. On the contrary, we actively select the uncertain compression policy for evaluation to obtain its actual accuracy, and train the performance predictor with the sampled policy that offers informative supervision. We first demonstrate the performance predictor learning with policy uncertainty, and then depict the active selection for uncertain policy.

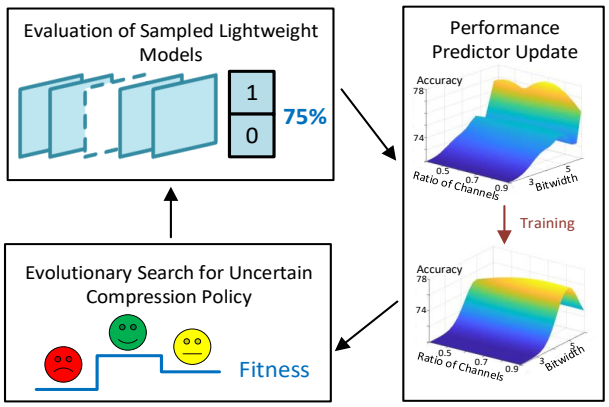


Fig. 3 The pipeline of learning the performance predictor via active compression policy evaluation, where we iteratively search the most uncertain compression policy defined by (15) via evolutionary algorithms, obtain the actual accuracy of sampled lightweight models via exhaustive training and update the performance predictor with the actual accuracy of sampled compression policy according to (12).

3.3.1 Performance Predictor Learning with Policy Uncertainty

Training the performance predictor via compression policies with uncertain prediction provides informative supervision, since the performance predictor obtains more knowledge from those samples (Beluch et al., 2018; Gal et al., 2017). Therefore, exhaustively training models compressed by those policies makes significant contribution to enhance the precision of the performance predictor. Figure 3 illustrates the pipeline of performance predictor learning in our SeerNet. For a given backbone, we iteratively search the uncertain compression policy via evolutionary algorithms, evaluate the sampled lightweight models to obtain the actual accuracy, and update the performance predictor with the actual accuracy of sampled compression policies. The well-trained performance predictor is employed for ultrafast compression policy optimization, so that flexible net-

work deployment with different resource constraint is achieved without complicated compression policy search and evaluation.

The influence of the compression policy perturbation on predicted performance reveals prediction uncertainty, where that sensitive to perturbation indicates highly uncertain prediction (Abbasnejad et al., 2020; Vijayanarasimhan and Grauman, 2014). Hence, the training loss of more uncertain compression policies should be weighted more greatly to strengthen the supervision informativeness. We employ the importance sampling by reweighting samples in the objective function $R(\boldsymbol{w})$ to train the performance predictor with the parameters $\boldsymbol{w}$ (Abbasnejad et al., 2020; Goyal et al., 2019):

$$\begin{aligned}\min_{\boldsymbol{w}} R(\boldsymbol{w}) &= \mathbb{E}_{\boldsymbol{s}\sim p(\boldsymbol{s})}\mathbb{E}_{a\sim p(a|\boldsymbol{s})} l(f(\boldsymbol{s}), a) \\ &= \mathbb{E}_{\boldsymbol{s}\sim p(\boldsymbol{s})}\mathbb{E}_{a\sim p(a|\hat{\boldsymbol{s}})} l(f(\boldsymbol{s}), a)\frac{p(a|\boldsymbol{s})}{p(a|\hat{\boldsymbol{s}})} \end{aligned} \quad (11)$$

where a means the actual accuracy and $\hat{\boldsymbol{s}}$ represents the perturbed counterpart of $\boldsymbol{s}$. $p(\boldsymbol{s})$ is the prior distribution of the compression policy. $p(a|\boldsymbol{s})$ and $p(a|\hat{\boldsymbol{s}})$ demonstrate the posterior distribution of accuracy given the compression policy $\boldsymbol{s}$ and the perturbed one $\hat{\boldsymbol{s}}$ respectively. $l(f(\boldsymbol{s}), a)$ is the loss function of accuracy prediction, which is defined as the mean squared error (MSE). In the importance sampling, the compression policy whose accuracy varies more significantly with the perturbation acquires larger weights in the learning objective. Since we leverage deterministic neural networks to predict the accuracy of various lightweight models, we optimize the following alternative objective $R^*(\boldsymbol{w})$ for the performance predictor, which is mathematically formulated in the appendix. The goal of (11) is to heavily weight the compression policy whose predicted accuracy is very different from the perturbed one, and we present the L_2 difference between predicted accuracies of the vanilla compression policy and the perturbed one as importance weights in the alternative objective:

$$\min_{\boldsymbol{w}} R^*(\boldsymbol{w}) = \sum_{i=1}^{N}\sum_{\hat{\boldsymbol{s}}^i}(f(\boldsymbol{s}^i) - a^i)^2 \cdot ||f(\boldsymbol{s}^i) - f(\hat{\boldsymbol{s}}^i)||_2^2 \quad (12)$$

where N is the number of actively sampled compression policies for performance predictor training. $\boldsymbol{s}^i$ and $\hat{\boldsymbol{s}}^i$ mean the i^{th} sampled compression policy and its perturbed counterpart. a^i represents the actual accuracy of the i_{th} compressed model obtained via exhaustively training. The L_2 difference of the predicted accuracy between the compression policy $\boldsymbol{s}^i$ and the perturbed counterparts reflects the importance weight. By penalizing the compression policy that is more sensitive to perturbation, the accurate performance predictor is learned by informative supervision with acceptable training cost.

Algorithm 2 Active performance predictor learning

Input: Backbone Network $\mathcal{N}$, the number of compression policy sampling K, performance predictor learning round Max_ro, evolution round Max_iter.
Output: Accurate performance predictor f^*.
Initialize: Randomly assign the weights of f.
for $t = 1, 2, ..., Max\_ro$ **do**
  Randomly sample K/Max_ro compression policy $\boldsymbol{s}$.
  for $i = 1, 2, ..., Max\_iter$ **do**
    Generate perturbed compression policy $\hat{\boldsymbol{s}}$ via (13).
    Predict the performance $f(\boldsymbol{s})$ and $f(\hat{\boldsymbol{s}})$.
    Select the top-k compression policy with the highest fitness according to (15).
    Mutation and crossover for the next generation $\boldsymbol{s}$.
  end for
  Train and validate $\mathcal{N}_{\boldsymbol{s}}$ for actual performance.
  Train f with the actual performance of $\boldsymbol{s}$ via (12).
end for
return the performance predictor f.

3.3.2 Active Selection for Uncertain Policy

In this section, we introduce the details for the search strategy of uncertain compression policies, which provides informative supervision for performance predictor learning. The uncertainty of the compression policy is evaluated by the accuracy sensitivity with respect to the perturbation on the policy space. The network capacity revealed by the complexity varies differently with the channel pruning ratio or the bitwidths of weights and activations across layers, because the pruning and quantization policies for different layers contribute diversely to the overall BOPs. Since the network capacity has significant influence on the model accuracy, the compression policy variation for uncertainty evaluation should enforce all perturbed counterparts to change the network capacity identically. Therefore, the uncertainty of different policies should be fairly estimated without the impacts of the network capacity variation.

Specifically, we generate each perturbed compression policy $\hat{\boldsymbol{s}}$ for the original one $\boldsymbol{s}$ by modifying one element with the following criteria:

$$\{\hat{\boldsymbol{s}} \big| |s_k - \hat{s}_k| = \alpha_m \cdot \mathbb{I}[k = m],\ k = 1, 2, ..., 3L\} \quad (13)$$

where s_k and $\hat{s}_k$ mean the k_{th} element of $\boldsymbol{s}$ and $\hat{\boldsymbol{s}}$, and the indicator function $\mathbb{I}[x]$ equals to one for true x and to zero otherwise. By varying $m \in \{1, 2, ..., 3L\}$, we acquire $3L$ perturbed policies that increase and decrease BOPs respectively, which result in $6L$ perturbed policies in total for the uncertainty evaluation of $\boldsymbol{s}$. Meanwhile, α_m is the scale coefficient to ensure the complex-

ity variation consistency for various perturbed policies:

$$\alpha_m = B_0 \cdot (\frac{\partial C(\boldsymbol{\mathcal{N}}_{\boldsymbol{s}})}{\partial s_m})^{-1} \tag{14}$$

where B_0 is a hyperparameter that demonstrates the model capacity variation for policy perturbation, and $\frac{\partial C(\boldsymbol{\mathcal{N}}_{\boldsymbol{s}})}{\partial s_m}$ depicts the sensitivity of the model complexity $C(\boldsymbol{\mathcal{N}}_{\boldsymbol{s}})$ defined in (3) with respect to the element s_m.

Because the compression policy with high prediction uncertainty contributes significantly in performance predictor learning according to (12), we sample the compression policies for actual performance acquisition via the following criteria in order to provide most informative supervision:

$$\boldsymbol{s} = \arg\max \sum_{\hat{\boldsymbol{s}}} ||f(\boldsymbol{s}) - f(\hat{\boldsymbol{s}})||_2 \tag{15}$$

Since the compression policy space is extremely large, we present the evolutionary search to select the most uncertain compression policy for the performance predictor training. In the evolutionary search, the genes are the vectors $\boldsymbol{s}$ representing the compression policy. We first randomly select the genes for initialization and obtain the their fitness $\mathcal{F}$ defined as $\mathcal{F} = \sum_{\hat{\boldsymbol{s}}} ||f(\boldsymbol{s}) - f(\hat{\boldsymbol{s}})||_2$. The top-k genes with the highest fitness are chosen for generating off-spring genes via the mutation and crossover process. The mutation process is carried out by randomly varying a proportion of elements that demonstrate the pruning ratio and quantization bitwidths in the genes, and the crossover process means that we recombine the pruning and quantization policies in two parent genes for off-spring generation. By iteratively selecting the top-k genes with the highest fitness and generating new genes with mutation and crossover, the compression policy with the most uncertain prediction is selected, which offers informative supervision for performance predictor learning. As the fitness of candidates can be evaluated by predicting the accuracy of the compression policies and their perturbed counterparts, the evolutionary search for the uncertain compression policies is computationally efficient. Algorithm 2 demonstrates the active performance predictor learning process, where the performance predictor is learned offline and utilized in ultrafast compression policy optimization for flexible deployment.

4 Experiments

In this paper, we conducted extensive experiments to evaluate our methods on the CIFAR-10 and ImageNet datasets for image classification and on the PASCAL VOC and COCO datasets for object detection. We first briefly introduce the datasets and the implementation details, and then verify the effectiveness of the presented ultrafast compression policy optimization and active compression policy evaluation for performance predictor learning via ablation study. Finally, we compare our SeerNet with the existing automated model compression methods to show our superiority.

4.1 Datasets and Implementation Details

We introduce the datasets we carried experiments on and data preprocessing techniques in the following:

CIFAR-10: The CIFAR-10 dataset includes $60,000$ samples with the resolution of 32×32, which are equally divided into 10 classes. We leveraged $50,000$ and $10,000$ images as the training and test sets respectively. We padded 4 pixels on each side of the images and randomly cropped them into the size of 32×32. Moreover, we scaled and biased all pixels into the range $[-1, 1]$.

ImageNet: ImageNet (ILSVRC2012) consists of approximately 1.2 million training and 50K validation images collected from $1,000$ categories. Following the data preprocessing techniques of bias extraction applied in CIFAR-10, we randomly cropped a 224×224 region from the resized image whose shorter side was 256 during the training process. For inference, we adopted a 224×224 center crop from the validation images.

PASCAL VOC: PASCAL VOC includes images from 20 different classes. Our model is trained on the VOC 2007 and VOC 2012 trainval sets consisting of about 16k images, and we evaluated our SeerNet on VOC 2007 test set containing around 5k images. Following (Everingham et al., 2010), we employed the mean average precision (mAP) as our evaluation criterion.

COCO: The images in the COCO dataset were collected from 80 different categories, and our experiments were conducted on the 2014 COCO object detection track. We trained our model with the combination of 80k images from the training set and 35k images selected from validation set (trainval35k (Bell et al., 2016)), and tested our SeerNet on the remaining minival validation set (Bell et al., 2016) including 5k images. Following the standard COCO evaluation metric (Lin et al., 2014), we apply the mean average precision (AP) for IoU $\in [0.5 : 0.05 : 0.95]$ as the evaluation metric. We also report average precision with the IOU threshold 50% and 75% represented as AP_{50} and AP_{75} respectively. Moreover, the average precision of small, medium and large objects notated as AP_s, AP_m and AP_l are also depicted.

For image classification, we employed architectures of VGG-small (Zhang et al., 2018) and ResNet20 (He et al., 2016) for automated model compression on CIFAR-10, and compressed ResNet18, ResNet50 and MobileNetV2

(Sandler et al., 2018) architectures with various computational cost constraint on ImageNet. For object detection, we adopted the SSD (Liu et al., 2016) framework with VGG16 and the Faster R-CNN (Ren et al., 2015) framework with ResNet18. Our performance predictor consisted of three fully-connected layers with the ReLU activation function. We iteratively trained the performance predictor with the accuracy of sampled lightweight models and actively searched uncertain compression policies via evolutionary algorithms.

In the sampled compression policies, the choices for the pruning ratio of all layers were $\{0.25, 0.5, 0.75\}$, while the selections for the weight and activation bitwidths of each layer were set as $\{2, 4, 6, 8\}$. Since quantizing the weights and activations in the first and last layers with low-precision significantly degrade the model performance, we set the bitwidth of the first and last layers of the backbone to 8 following (Wang et al., 2019a). For perturbed compression policy generation, the hyperparameter B_0 was positively related to the original BOPs of the full-precision networks. Varying each element in the compression policies yields $6L$ perturbation for networks with L layers, and we only randomly sampled $0.5L$ perturbed compression policies for performance prediction to reduce the computational cost in uncertainty estimation. We trained 800 lightweight models with different compression policies for accuracy acquisition in order to learn the performance predictor of each backbone and dataset. 50 compressed models were actively sampled for evaluation and performance predictor training in each round out of 16 rounds, where the compression policies that initially trained the performance predictor were randomly selected. We set the population size to be 100 in the evolutionary search for uncertain compression policies, where the top-25 candidates based on (15) produced the next generation. 50 candidates randomly mutated with the mutation rate 0.1 for the compression policy of each layer. For crossover, the compression policy of each layer was randomly chosen from 50 parent candidates. The max iterations were 500 for the best candidate selection.

In compression policy optimization, the hyperparameters λ_1 and λ_2 in the objective were 0.1 and 0.005, and the hyperparameters μ for gradient accumulation and η for adaptive stepsizes were 0.9 and 0.05. The maximum iteration step for updating the compression policy was 30. We randomly selected compression policy whose complexity was approximately half of the computational cost constraint for initialization, and updated the compression policies until reaching the maximum iteration or the computational cost constraint.

We employed the max response selection (Han et al., 2015b) that pruned weights according to the magnitude for channel pruning. Meanwhile, we followed the implementation in (Wang et al., 2019a) for weight and activation quantization. During training of the lightweight networks, we used the Adam optimizer (Kingma and Ba, 2014) with the batchsize of 256. For CIFAR-10, we initialized the learning rate as 0.001 and decayed twice at the 60_{th} and 80_{th} training epochs out of 100 epochs, where the learning rate multiplied 0.1 for each decay. For ImageNet, the learning rate started from 0.005 and decayed at the 20_{th} and 40_{th} in the total 60 epochs with the same decay rate. The backbone for object detection was pretrained on ImageNet following the above implementation details. For the network finetuning on object detection, the learning rate was initially set as $1e$-3 and decreased to $1e$-4 and $1e$-5 at the 40_{th} and 60_{th} epoch out of 80 epochs for PASCAL VOC, and started from 0.001 with the same decay strategy at the 6_{th} and 10_{th} epoch during 12 training epochs for COCO.

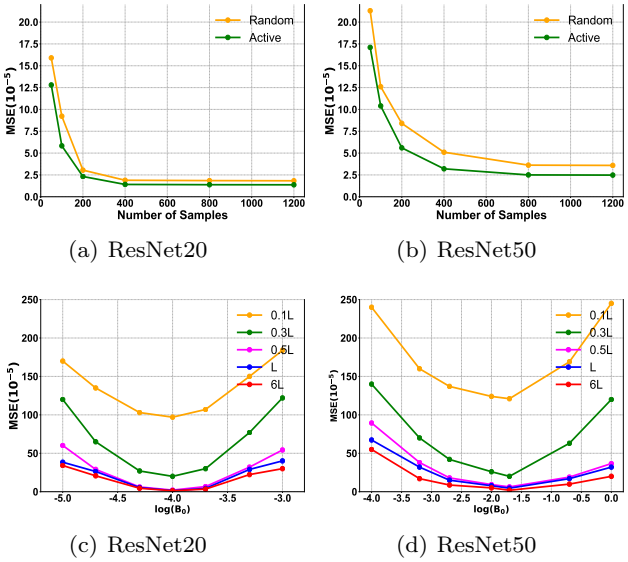
4.2 Ablation Study

To verify the benefits of active compression policy evaluation for performance predictor learning, we conducted the ablation study to assess our performance predictor w.r.t. different sampling strategies for compression policies and various numbers of sampled lightweight models on ResNet20 and ResNet50. With the same architectures, we varied the perturbation magnitude in uncertainty estimation with different numbers of perturbed compression policies in order to show the influence.

For the ablation study of compression policy optimization with ResNet20, we validate the effectiveness and efficiency by comparing the accuracy-complexity trade-off with the optimal lightweight models obtained via other search strategies including reinforcement learning and evolutionary algorithms. In order to verify the impact of the barrier complexity loss, the rounded policy gap minimization, the momentum-based policy update and adaptive stepsizes in our ultrafast compression policy optimization, we report the accuracy-complexity trade-off of compression policies obtained via different combinations of the above techniques. Moreover, we investigate the impact of the initialization and the stepsize scale of policy update. The ablation study was conducted on CIFAR-10 with the BOPs constraint 0.2G and 0.4G for ResNet20 and ResNet50 respectively.

4.2.1 Effects of Performance Predictor Learning

Performance w.r.t. different sampling strategies and varying numbers of sampled lightweight models: We trained the predictor with the actual accuracies of 50, 100, 200, 400, 800 and $1,200$ compressed



(a) ResNet20 (b) ResNet50

(c) ResNet20 (d) ResNet50

Fig. 4 (a) and (b) report the MSE ($\times 10^{-5}$) between the predicted and actual accuracy of the sampled compression policies with different sampling strategies and varying numbers of samples for ResNet20 and ResNet50 architectures respectively. (c) and (d) show that with various perturbation magnitude and different numbers of sampled perturbed policies in uncertainty estimation for ResNet20 and ResNet50.

models obtained via random and active compression policy evaluation, where 50 compression policies were also randomly sampled for validation. We depict the MSE between the actual and predicted accuracies in Figure 4 (a) and (b) for ResNet20 and ResNet50. Our active sampling strategy chooses the uncertain compression policies that provide informative supervision for performance predictor learning, so that the predicted accuracy is more precise compared with the random sampling strategy. The advantages are more obvious for small training sets, which reveals the benefits of our active sampling for automated model compression in extremely low search cost. Training the performance predictor for ResNet50 requires more sampled policies to achieve low prediction error due to the larger search space. However, sampling more compression policies only slightly influences the MSE between the actual and predicted accuracies when the training set exceeds 800 samples for the ResNet50 architecture, and we evaluated 800 compression policies to learn the performance predictor in other experiments. The actual and predicted accuracies of randomly and actively sampled policies on different datasets across various network architectures are demonstrated in the appendix.

Impacts of perturbation magnitudes and the number of sampled perturbed policies in uncertainty estimation: The hyperparameter B_0 represents the model capacity changes caused by perturbed compression policies, and we employed different settings for B_0 and show the influence on performance predictor learning. Given the perturbation magnitude of model capacity, we randomly sampled various numbers of perturbed compression policies and report the MSE between predicted and actual accuracies. 50 randomly sampled compression policies were utilized for validation. Figure 4 (c) and (d) demonstrate the results on ResNet20 and ResNet50 respectively. Medium B_0 results in the minimal MSE in both architectures, as small perturbation fails to collect sufficient information for uncertainty estimation and large one considers non-local information that has little contribution to uncertainty. Meanwhile, the optimal perturbation magnitude in ResNet50 is larger than ResNet20 due to the higher original model complexity of backbone networks. Sampling more perturbed compression policies positively contributes to the precision of the performance predictor learning because of more accurate uncertainty estimation. However, sampling over $0.5L$ perturbation for each policy only slightly improves the uncertainty estimation, while the computational cost increases significantly. To maintain high computational efficiency, we randomly sampled $0.5L$ perturbation of each policy for uncertainty estimation in the rest experiments.

Table 1 The accuracy on CIFAR-10, computational complexity and the search cost (GPU hours) of the optimal compression policy obtained by reinforcement learning, evolutionary algorithms and our compression policy optimization (CPO) under the computational cost constraint BOPs less than 0.2G for ResNet20, where the reward for agents in reinforcement learning and fitness for population in evolutionary algorithms based on accuracy and model complexity were obtained via the learned performance predictor. The training cost is 0.58 GPU hours for the lightweight models.

	Acc.(%)	BOPs(G)	Cost
Reinforcement learning	92.03	0.19	0.34
Evolutionary algorithms	92.10	0.19	0.50
CPO	92.51	0.20	0.003

4.2.2 Effects of Compression Policy Optimization

Comparison with other search strategies: To validate the effectiveness and the efficiency of our ultrafast compression policy optimization, we compare the accuracy and the computational complexity of the optimal lightweight models searched by reinforcement learning and evolutionary algorithms, where the reward for agents in reinforcement learning and the fitness for population in evolutionary algorithms were obtained via the learned performance predictor. For reinforcement learning, we modified the implementations in (Wang et al., 2019a) by adding the pruning ratio in the state and action space. For evolutionary algorithms, we leveraged the pipeline in (Wang et al., 2020a) where the branch of architecture search was removed. The detailed imple-

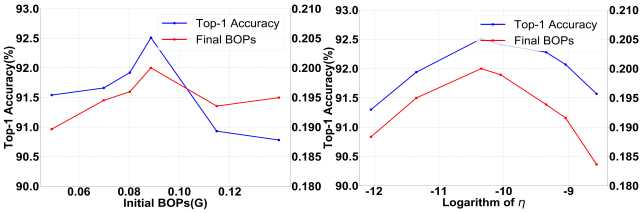
Table 2 The BOPs(G), compression ratio and the accuracy on CIFAR-10 of the obtained lightweight ResNet20 architectures. The existence of barrier complexity loss (Bar.) and the rounded gap minimization (Gap.) in the learning objective was varied. F\M, F&M, A\M and A&M respectively stand for fixed stepsizes without momentum, fixed stepsizes with momentum, adaptive stepsizes without momentum and adaptive stepsizes with momentum.

Bar.	Gap.	F\M			F&M			A\M			A&M		
		BOPs	Comp.	Top-1	BOPs	Comp.	Top-1	BOPs	Comp.	Top-1	BOPs	Comp.	Top-1
×	×	0.193	216.58	88.41	0.188	222.34	88.58	0.197	212.43	90.39	0.196	213.27	90.63
	✓	0.183	227.47	89.10	0.192	217.58	89.25	0.196	213.27	91.19	0.198	211.11	91.38
✓	×	0.195	214.35	88.23	0.193	216.58	88.79	0.191	218.85	90.94	0.192	217.71	91.22
	✓	0.198	211.11	89.35	0.197	212.43	89.64	0.200	209.00	91.75	0.200	209.00	**92.51**

mentations of reinforcement learning and evolutionary algorithms are demonstrated in the appendix. Table 1 demonstrates the accuracy, model complexity and the search cost of different search algorithms. Our ultrafast compression policy optimization acquires highest accuracy within the computational complexity constraint, and the search cost can be negligible compared with reinforcement learning and evolutionary algorithms.

Performance w.r.t. different terms in learning objectives and various techniques in policy update: Table 2 demonstrates the BOPs, compression ratio and accuracy of obtained light-weight networks with various objectives and update techniques. The existence of barrier complexity loss and the rounded gap minimization in the learning objectives was varied. The impacts of the presented adaptive stepsizes and the momentum-based gradient in compression policy update were also investigated. Comparing the model complexity and the accuracies across different rows, we conclude that more computational resource under the constraint is utilized with accuracy improvement via the barrier complexity loss. Meanwhile, the rounded policy gap minimization shrinks the difference between the optimal policy and the discrete one for deployment, and better accuracy-complexity trade-off is achieved. Comparing the performance across various columns, we observe that the adaptive stepsizes yield lightweight models with better performance within the expected complexity because of stable optimization process. Moreover, the gradient momentum provides historical information of the optimization process so that the obtained compression policy can escape the local maximum.

Performance w.r.t. different compression policy initialization: To investigate the influence of the compression strategy initialization on the performance of our compression policy optimization, we show the actual accuracies and the computation complexity of the optimal lightweight models w.r.t. the complexity of initialized compression policy in Figure 5 (a). The results show that medium complexity for initialized compressed models acquires the highest accuracy given the complexity constraint. High complexity for initialized lightweight models attains local minimum during the compression policy update, while low complexity for initialization cannot converge to the optimal compressed models before reaching the complexity constraint.



(a) Varying initialization (b) Varying stepsize scale

Fig. 5 The actual accuracy and computation complexity of the optimal lightweight models obtained by our compression policy optimization with (a) different initialization and (b) various update stepsize scale under the computational cost constraint BOPs less than 0.2G, where the ResNet20 architecture on CIFAR-10 was evaluated.

Performance w.r.t. various update stepsize scale η: The update stepsize scale is controlled by the hyperparameter η in (10), where low η generally leads to small stepsizes and vice versa. Figure 5 (b) depicts the actual accuracy and the computational complexity of the optimal compressed models acquired via different parameter settings of η. Medium stepsizes outperform other choices. Small stepsizes fail to achieve the optimal compression policy when reaching the maximum update iterations due to the local maximum, and large stepsizes enforce ultrafast compression policy optimization to be hard to converge.

4.3 Comparison with the State-of-the-art Methods

We compare our SeerNet with the fixed-precision quantization methods including LQ-Nets (Zhang et al., 2018), APoT (Li et al., 2020b), RQ (Louizos et al., 2018), LSQ (Esser et al., 2019), AdaBits (Jin et al., 2020), BitMixer (Bulat and Tzimiropoulos, 2021) and mixed-precision quantization approaches such as ALQ (Qu et al., 2020), DQ (Uhlich et al., 2019), BP-NAS (Yu et al., 2020), HAQ (Wang et al., 2019a), HMQ (Habi et al., 2020), HAWQ (Dong et al., 2019). Meanwhile, we compare the accuracy with the state-of-the-art automated model compression method DJPQ (Wang et al., 2020b) where the pruning and quantization policies were jointly searched. In order to show the performance in different accuracy-

Table 3 Comparison of network complexity, classification accuracy and search cost on CIFAR-10 with state-of-the-art network compression methods in VGG-small and ResNet20. W/A means the bitwidth of weights and activations respectively, and the computational complexity is measured by MACs (G) and BOPs (G). Comp. represents the network compression ratio calculated by the BOPs, and Acc. means the accuracy on image classification. The search cost is demonstrated by GPU hours, where N means the number of deployment scenarios. The number in the bracket of search cost for mixed-precision quantization demonstrates the break-even point of baseline methods whose search cost is higher than our SeerNet. The training cost for each model of VGG-small and ResNet20 is 0.43 and 0.58 GPU hours respectively.

Methods	W/A	MACs	BOPs	Comp.	Acc.	Cost
VGG-small						
Baseline	32/32	0.494	506.0	–	92.80	–
LQ-Nets	4/4	0.494	7.91	64.00	92.12	–
ALQ	mixed	0.494	7.44	68.00	90.90	$1.3N(11)$
SeerNet	mixed	0.459	7.35	68.84	92.97	$13+0.003N$
DQ*	mixed	0.613	3.40	185.00	91.59	$0.9N(15)$
SeerNet	mixed	0.340	3.88	130.41	92.84	$13+0.003N$
DJPQ*	mixed	0.367	2.99	210.37	91.54	$0.5N(27)$
SeerNet	mixed	0.223	2.27	222.91	92.69	$13+0.003N$
ResNet20						
Baseline	32/32	0.041	41.8	–	92.51	–
APoT	4/4	0.041	0.65	64.00	92.45	–
HMQ	mixed	0.041	0.65	64.00	92.59	$2.0N(11)$
SeerNet	mixed	0.038	0.63	65.02	92.71	$21+0.003N$
BP-NAS	mixed	0.041	0.39	106.35	92.12	$1.1N(20)$
SeerNet	mixed	0.030	0.34	122.94	92.55	$21+0.003N$
BP-NAS	mixed	0.041	0.32	128.97	92.04	$1.1N(20)$
SeerNet	mixed	0.038	0.20	209.00	92.51	$21+0.003N$

complexity trade-offs, we leveraged three BOPs constraints for each architecture. The reduction in MACs demonstrates the network pruning ratio, and the BOPs decrease reveals the total compression effect. Therefore, we define the reduction ratio of BOPs as the compression ratio. The reported search cost only contains computational cost to obtain the optimal compression policy, and that of baseline methods is evaluated by rerunning the released code or our re-implementation. The total cost for model deployment can be easily calculated by summing the search cost and the training cost. The break-even points indicates the number of scenarios where the search cost is higher than our SeerNet. The acquired compression policy of our SeerNet for different architectures is visualized in our appendix.

4.3.1 Comparison on Image Classification

Comparision on CIFAR-10: Table 3 shows the experimental results on CIFAR-10 with VGG-small and ResNet20. $*$ represents the methods utilizing the VGG7 architecture which is very similar to VGG-small. The fixed-precision quantization ignores the importance variety among different layers and fails to effectively assign the optimal bitwidth for each layer. The mixed-precision quantization does not consider the redundancy in different channels while the pruning strategy can further enhance the model efficiency. Compared with the state-of-the-art mixed-precision networks BP-NAS, our SeerNet enhances the accuracy by 0.47% (92.51% vs. 92.04%) with 1.60× BOPs reduction (0.20G vs. 0.32G) with the ResNet20 architecture. Although the automatic model compression method DJPQ jointly searches the pruning and quantization policy, the search deficiency leads to heavy computational cost due to the extremely large search space. On the contrary, our SeerNet obtains the optimal lightweight model without complex compression policy search and evaluation, so that the proposed method only requires 0.003 GPU hour to marginally search an automated model compression policy on both VGG-small and ResNet20. Because the number of deployment scenarios is usually very large in realistic applications with frequent changes of hardware configurations and battery levels, the compression policy search cost is reduced sizably in our SeerNet.

Table 4 The BOPs(G), top-1 classification accuracy and search cost on ImageNet with state-of-the-art network compression methods in MobileNet-V2, ResNet18 and ResNet50. The training cost for each model of MobileNet-V2, ResNet18 and ResNet50 is 37.4, 60.8 and 80.9 GPU hours respectively.

Methods	W/A	MACs	BOPs	Comp.	Top-1	Cost
MobileNet-V2						
Baseline	32/32	0.33	337.9	–	71.72	–
RQ	6/6	0.33	11.88	28.44	68.02	–
HMQ	mixed	0.33	10.97	30.80	71.40	$31.4N(24)$
SeerNet	mixed	0.25	10.82	31.22	71.47	$750+0.004N$
HAQ	mixed	0.33	8.25	40.96	69.45	$51.1N(15)$
DJPQ	mixed	0.28	7.87	42.96	69.30	$12.2N(62)$
SeerNet	mixed	0.22	7.69	43.91	70.76	$750+0.006N$
HMQ	mixed	0.33	5.25	64.40	70.90	$33.5N(23)$
DQ	mixed	0.33	4.92	68.67	69.74	$21.6N(35)$
SeerNet	mixed	0.19	4.88	69.26	70.55	$750+0.006N$
ResNet18						
Baseline	32/32	1.81	1853.4	–	69.74	–
ALQ	mixed	1.81	58.50	31.68	67.70	$34.7N(15)$
SeerNet	mixed	1.37	56.94	32.55	69.72	$500+0.003N$
DJPQ	mixed	1.39	35.01	52.94	69.27	$18.2N(28)$
HAWQ	mixed	1.81	34.00	54.51	68.45	$22.7N(23)$
SeerNet	mixed	1.22	31.84	58.21	69.48	$500+0.004N$
LSQ	3/2	1.81	10.86	170.66	66.90	–
BitMixer	2/2	1.81	7.24	256.00	64.40	–
ALQ	mixed	1.81	7.24	256.00	66.40	$38.5N(13)$
SeerNet	mixed	0.70	7.19	257.71	67.84	$500+0.004N$
ResNet50						
Baseline	32/32	3.86	3952.6	–	76.40	–
HAQ	mixed	3.86	94.92	41.64	75.30	$67.2N(15)$
SeerNet	mixed	3.34	91.99	42.97	76.70	$950+0.005N$
HAWQ	mixed	3.86	61.29	64.49	75.48	$34.5N(28)$
LQ-Net	4/4	3.86	61.76	64.00	75.10	–
AdaBits	4/4	3.86	61.76	64.00	76.10	–
SeerNet	mixed	2.97	59.66	66.25	76.61	$950+0.006N$
AdaBits	3/3	3.86	34.74	113.78	75.80	–
HMQ	mixed	3.86	37.72	104.8	75.45	$49.4N(20)$
BP-NAS	mixed	3.86	33.22	118.98	75.71	$35.6N(27)$
SeerNet	mixed	2.12	31.67	124.81	75.90	$950+0.007N$

Comparison on ImageNet: The results on ImageNet with MobileNetV2, ResNet18 and ResNet50 are demonstrated in Table 4. Mixed-precision quantization

Table 5 Comparison of BOPs(G), mean average precision and search cost on PASCAL VOC with existing network compression methods, where the SSD framework with VGG16 and Faster R-CNN with ResNet18 was employed. The training cost for each model of VGG16 and ResNet18 is 19.5 and 18.7 GPU hours respectively.

Methods	W/A	MACs	BOPs	Comp.	mAP	Cost
SSD & VGG16						
Baseline	32/32	27.14	27787.7	–	72.4	–
HAQ	mixed	27.14	846.67	32.82	68.9	$48.8N(17)$
SeerNet	mixed	18.23	768.25	36.17	69.7	$800{+}0.005N$
DJPQ	mixed	16.11	627.12	44.31	66.8	$29.1N(28)$
SeerNet	mixed	14.68	622.76	44.62	68.8	$800{+}0.005N$
BP-NAS	mixed	27.14	453.60	61.26	66.7	$33.2N(25)$
SeerNet	mixed	13.59	435.27	63.84	67.3	$800{+}0.006N$
Faster R-CNN & ResNet18						
Baseline	32/32	22.01	22534.8	–	74.5	–
APoT	5/5	22.01	550.17	40.96	71.2	–
SeerNet	mixed	19.04	526.39	42.81	73.4	$650{+}0.005N$
HMQ	mixed	22.01	343.41	65.62	71.3	$28.9N(23)$
DJPQ	mixed	17.19	349.54	64.47	69.7	$27.2N(24)$
SeerNet	mixed	15.21	339.64	66.35	72.3	$650{+}0.005N$
BP-NAS	mixed	22.01	298.63	75.46	69.7	$22.4N(30)$
SeerNet	mixed	13.22	298.43	75.51	71.8	$650{+}0.006N$

outperforms fixed-precision quantization by a larger margin on ImageNet compared with CIFAR-10 as the optimal bitwidth assignment makes more contribution to the image classification on challenging datasets. Compared with the state-of-the-art mixed-precision quantization method BP-NAS, our SeerNet improves the top-1 accuracy by 0.19% (75.90% vs. 75.71%) and decreases BOPs by 1.05× (31.67G vs. 33.22G) with the ResNet50 architecture due to the automated pruning strategy. Meanwhile, the marginal search cost is decreased by 5086× (35.6 GPU hours vs. 0.007 GPU hours). Compared with the state-of-the-art automatic model compression method, our SeerNet directly optimizes the compression policy with the discriminative performance predictor without resource-exhaustive compression policy search and evaluation process. We obtain better accuracy-complexity trade-off with only 0.05% and 0.02% marginal search cost compared with DJPQ in MobileNet-V2 and ResNet18 respectively. The marginal cost reduction is much more sizable on ImageNet compared with CIFAR-10 due to the significantly increased training cost for each sampled compression policy in exhaustive evaluation, which shows the superiority of our SeerNet in deployment facing largescale datasets.

4.3.2 Comparison on Object Detection

Comparison on PASCAL VOC: Table 5 illustrates the computational complexity and the mAP of different compression methods on PASCAL VOC, where the search cost does not contain the computational cost for the model pretraining on ImageNet. Similar to image classification, the mixed-precision networks achieve more optimal accuracy-complexity trade-offs with different computational cost constraints. Our SeerNet enhances the mAP of the state-of-the-art DJPQ by 2.6% (72.3% vs. 69.7%) in the Faster R-CNN framework with VGG16, where the BOPs are similar. Moreover, SeerNet only requires 0.005 GPU hour compared with 27.2 GPU hours in DJPQ for marginal compression policy search. The significant search efficiency improvement enables flexible model deployment for different hardware configurations and battery levels in realistic applications relying on object detection such as autonomous driving (Chen et al., 2017), which usually requires models with hundreds of complexity constraints.

Comparison on COCO: Despite of the BOPs and the mAP on COCO, we also show the average precision at different IoU thresholds and that for objects in various sizes. Table 6 depicts the results, where our SeerNet achieves better accuracy-complexity trade-offs than conventional mixed-precision networks and automatic model compression methods across different detection frameworks and backbone architectures. SeerNet is free of complex policy search and evaluation stage, and we only require 0.02% search cost (0.006 GPU hours vs. 38.7 GPU hours) to marginally acquire the promising pruning and quantization policy for the Faster R-CNN detector with ResNet18 backbone in flexible deployment. Since training deep neural networks on the largescale COCO dataset costs much more computational resources, our SeerNet saves the computational cost of optimal compression policy acquisition more sizably compared with that trained on PASVAL VOC.

5 Conclusion

In this paper, we have presented the ultrafast automated model compression framework for flexible network deployment. The proposed SeerNet learns the accurate performance predictor in acceptable training cost via active compression policy evaluation, where the most uncertain pruning and quantization strategies with informative supervision are selected by efficient evolutionary search. Then the gradient that maximizes the predicted performance under the barrier complexity constraint is leveraged to differentiably search the desirable compression policy, where adaptive update step-sizes with momentum are employed to strengthen the optimality of the acquired pruning and quantization strategies. Therefore, ultrafast automated model compression is achieved without resource-exhaustive compression policy search and evaluation. Extensive experiments on image classification and object detection demonstrate the superiority in efficiency and effectiveness of the proposed method. There are two interesting

Table 6 Comparison of BOPs(G), mAP@[.5, .95] and search cost on COCO with state-of-the-art network compression methods in the SSD framework with VGG16 and Faster R-CNN with ResNet18. The average precision at different IoU thresholds and that for objects in various sizes are also illustrated. The training cost of SSD and Faster R-CNN is 56.5 and 53.2 GPU hours.

Methods	W/A	MACs	BOPs	Comp.	mAP	AP_{50}	AP_{75}	AP_s	AP_m	AP_l	Cost
SSD & VGG16											
Baseline	32/32	27.14	27787.7	–	23.2	41.2	23.4	5.3	23.2	39.6	–
DJPQ	mixed	18.39	795.98	34.91	20.1	37.9	20.6	5.3	22.3	34.7	$53.6N(26)$
SeerNet	mixed	21.90	783.86	35.45	22.7	40.2	22.8	5.8	24.4	35.9	$1350+0.005N$
BP-NAS	mixed	27.14	616.14	45.10	20.8	38.4	20.7	5.2	22.5	33.8	$42.6N(32)$
SeerNet	mixed	19.03	569.89	48.76	22.1	39.9	22.6	6.0	24.0	36.1	$1350+0.005N$
HAQ	mixed	27.14	445.67	62.35	20.1	37.5	19.9	5.2	21.3	32.6	$95.3N(15)$
APoT	4/4	27.14	434.18	64.00	18.1	34.4	17.5	4.5	19.1	29.4	–
SeerNet	mixed	15.51	421.73	65.89	21.3	39.0	21.3	5.7	23.5	34.2	$1350+0.006N$
Fatser R-CNN & ResNet18											
Baseline	32/32	22.01	22534.8	–	26.0	44.8	27.2	10.0	28.9	39.7	–
HAQ	mixed	22.01	471.83	47.76	25.5	44.0	26.3	12.8	27.5	33.8	$89.9N(14)$
SeerNet	mixed	19.31	458.93	49.10	26.8	45.7	28.1	13.8	29.3	35.0	$1200+0.005N$
DJPQ	mixed	18.24	342.32	65.83	24.4	40.4	25.6	11.4	25.6	30.7	$47.3N(26)$
APoT	4/4	22.01	352.11	64.00	23.2	39.9	24.1	11.6	25.3	30.1	–
SeerNet	mixed	14.25	327.59	68.79	25.5	44.4	26.3	12.6	27.9	33.8	$1200+0.005N$
HMQ	mixed	22.01	301.23	74.81	24.1	42.8	24.5	12.7	26.9	30.4	$55.5N(22)$
BP-NAS	mixed	22.01	312.38	72.14	23.6	41.9	23.9	12.7	27.6	32.0	$38.7N(32)$
SeerNet	mixed	12.39	282.60	79.74	25.1	43.7	25.9	13.8	28.2	32.9	$1200+0.006N$

directions for the future work: (1) extending our SeerNet to other network architectures such as transformers and graph neural networks, (2) implementing the SeerNet method with hardware cost constraint such as latency and energy.

Acknowledgments

This work was supported in part by the National Key Research and Development Program of China under Grant 2017YFA0700802, in part by the National Natural Science Foundation of China under Grant 62125603, and in part by a grant from the Beijing Academy of Artificial Intelligence (BAAI).

Compliance with Ethical Standards

Conflict of interest The authors declare that they have no conflict of interest.
Ethical approval This article does not contain any studies with human participants or animals.

Appendix

A. Mathematical formulation from (11) to (12) of the manuscript.

Since we leverage the deterministic neural networks to predict the accuracy of various lightweight models, we present the alternative objective function to tractably calculate the objective (11) in the manuscript based on importance sampling. We first rewrite the importance weight in (11) of the manuscript via the analytical form of Dirac-delta function:

$$\frac{p(a|\boldsymbol{s})}{p(a|\hat{\boldsymbol{s}})} = \frac{\lim\limits_{\epsilon\to 0}\frac{1}{\pi}\frac{\epsilon}{\epsilon^2+(a-a_{10})^2}}{\lim\limits_{\epsilon\to 0}\frac{1}{\pi}\frac{\epsilon}{\epsilon^2+(a-a_{20})^2}} \tag{16}$$

where a_{10} and a_{20} are distribution parameters of original and perturbed policies parameterized by the performance predictor. As ϵ is higher-order infinitesimal of $a-a_{10}$ and $a-a_{20}$ according to the definition of Dirac-delta function, the importance weight can be rewritten as follows:

$$\begin{aligned}\frac{p(a|\boldsymbol{s})}{p(a|\hat{\boldsymbol{s}})} &= \frac{(a-a_{20})^2}{(a-a_{10})^2} \\ &= \frac{(a-a_{10})^2+(a_{10}-a_{20})^2+2(a-a_{10})(a_{10}-a_{20})}{(a-a_{10})^2}\end{aligned}$$

Since a means the predicted accuracy parameterized by a_{10}, the difference between a and a_{10} can be assumed as a small constant γ_{10} in the deterministic settings. Therefore, the difference between a and a_{10} is far less than that between a_{10} and a_{20}. We obtain the following relationship between $\frac{p(a|\boldsymbol{s})}{p(a|\hat{\boldsymbol{s}})}$ and $a_{10}-a_{20}$:

$$\frac{p(a|\boldsymbol{s})}{p(a|\hat{\boldsymbol{s}})} = (a_{10}-a_{20})^2/\gamma_{10} \propto (a_{10}-a_{20})^2 \tag{17}$$

a_{10} and a_{20} are represented by $f(\boldsymbol{s})$ and $f(\hat{\boldsymbol{s}})$, and the loss function $l(f(\boldsymbol{s}), a)$ of accuracy prediction is assigned with the L_2 norm of the difference between the predicted and actual accuracy $(f(\boldsymbol{s}^i)-a^i)^2$. Therefore, we optimize the alternative objective (12) in the manuscript to provide informative supervision for performance predictor learning.

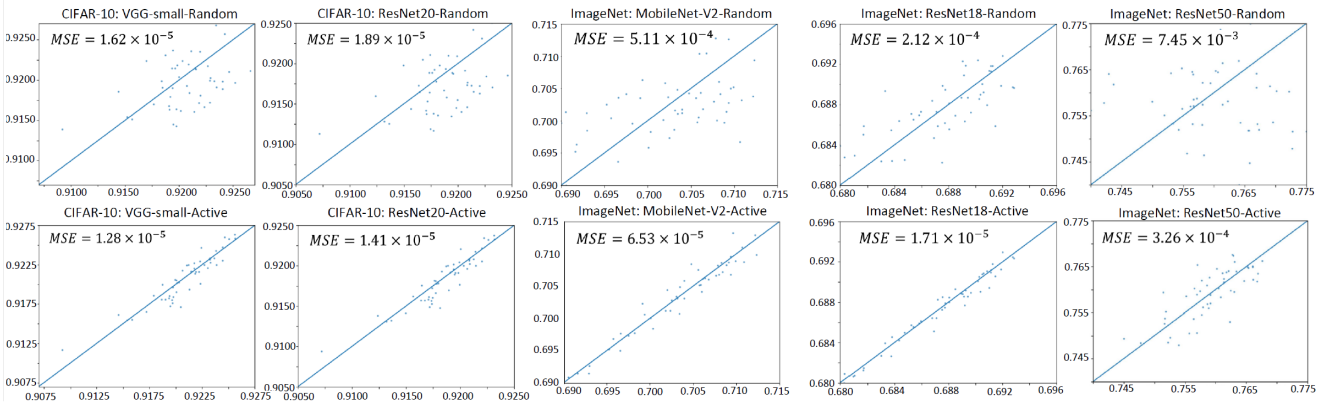


Fig. 6 The actual and predicted accuracy for different compression policy across various datasets and architectures, where random sampling and our active sampling are both evaluated. The architectures for evaluation on CIFAR-10 include VGG-small and ResNet20, and those on ImageNet contain MobileNet-V2, ResNet18 and ResNet50. The horizontal axis represents the predicted accuracy and the vertical axis means the actual accuracy. The mean squared errors (MSE) are also demonstrated in the figure.

Table 7 The accuracy(%) and BOPs(G) variance for Table 1 in the manuscript by running experiments for 5 times.

	Acc.(%)	BOPs(G)
Reinforcement learning	91.95 ± 0.21	0.19 ± 0.01
Evolutionary algorithms	92.08 ± 0.05	0.19 ± 0.01
CPO	92.38 ± 0.17	0.20 ± 0

B. The Actual and Predicted Accuracies for Sampled Lightweight Networks

We employed the architectures of VGG-small (Zhang et al., 2018) and ResNet20 (He et al., 2016) for automated model compression on CIFAR-10, and compressed MobileNet-V2 (Sandler et al., 2018), ResNet18 and ResNet50 architectures on ImageNet. We trained the performance predictor by the accuracy of 800 randomly sampled and 800 actively sampled compressed models respectively, and regressed the accuracy for 50 randomly sampled lightweight models via the well-trained performance predictor. We show the actual and predicted accuracy of random sampling and our active sampling in Figure 6, where the MSE between the actual and predicted accuracy is also demonstrated. The predicted accuracy is generally closed to the actual one across the datasets, which shows the effectiveness of the performance predictor for automated model compression. Meanwhile, our active sampling strategy chooses the uncertain compression policies that provide informative supervision for performance predictor learning, so that the predicted accuracy is more precise compared with the random sampling strategy. Since the performance variance for different quantization and pruning strategies on largescale datasets is larger, and our active sampling strategy offers more benefits for the performance predictor learning on ImageNet. Although deeper architectures with large search space such as MobileNet-V2 and ResNet50 obtain higher MSE for their performance predictors, the active sampling policy is still capable of providing informative supervision for accurate performance predictor learning as the MSE is less than 5×10^{-4}.

C. Visualization of the Optimal Compression Policy

We show the bitwidth of weights and activations and pruning ratio across different layers for image classification in Figure 7, where the computational cost constraint is 2.4G and 0.2G BOPs for compressing VGG-small and ResNet20 on CIFAR-10 and is 8G, 33G and 62G BOPs for MobileNet-V2, ResNet18 and ResNet50 compression on ImageNet respectively.

Because sparse networks require precise weights and activations to maintain the representational capacity and increasing the bitwidth of layers with high pruning ratio only brings slight computational cost, the layers with high pruning ratio are usually assigned with large bitwidth in the optimal compression policy. The optimal compression policy searched via the state-of-the-art method DJPQ (Wang et al., 2020b) only prunes the bottom layers since they sequentially prune the networks from bottom layers to top layers, while our SeerNet simultaneously optimizes the pruning strategy for all layers and preserves the informative channels with redundant channel removal.

For VGG-small and ResNet20 architectures trained on CIFAR-10, the bitwidth varies slightly across different layers. Meanwhile, the pruning ratio is high and the bitwidth is low for all layers, which means significant over-parameterization for both network architec-

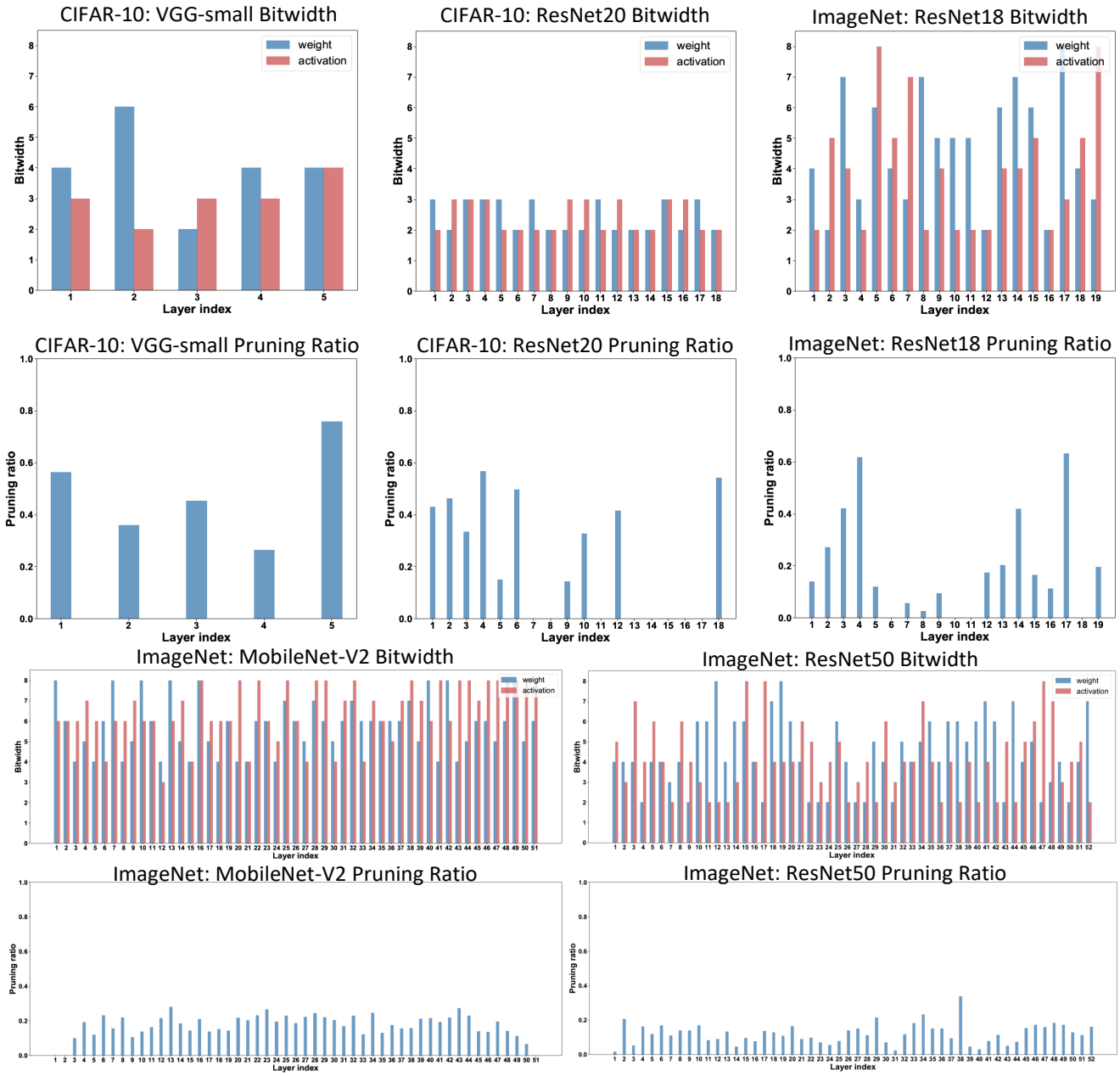


Fig. 7 The visualization of the optimal compression policies obtained by our ultrafast SeerNet with given computational cost constraint on image classification. We utilized VGG-small and ResNet20 architectures on CIFAR-10 and MobileNet-V2, ResNet18 and ResNet50 networks on ImageNet.

tures on CIFAR-10. The large bitwidth and low pruning ratio in the optimal compression policy for MobileNet-V2 indicates that the compact architecture is hard to be further compressed without sizable accuracy drop. On the contrary, ResNet50 is compressed with extremely low bitwidth, which demonstrates the significant redundancy. Moreover, the bitwidth of activations is usually larger than that of weights, which depicts that the model accuracy is more sensitive to activation quantization than weight quantization.

D. Implementation Details in Section 4.2.2

To validate the effectiveness of the presented performance predictor, we utilized the reinforcement learning and evolutionary algorithms to search the optimal compression policy via our performance predictor. For reinforcement learning, we leveraged the deep deterministic policy gradient (DDPG) (Lillicrap et al., 2015). Compared with (Wang et al., 2019a), we added an extra state p_k representing the pruning ratio for the k_{th} con-

Table 8 The accuracy(%) and BOPs(K) variance for Table 2 in the manuscript acquired by running experiments for 5 times.

Bar.	Gap.	F\M		F&M		A\M		A&M	
		BOPs	Top-1	BOPs	Top-1	BOPs	Top-1	BOPs	Top-1
×	×	193 ± 1	88.10 ± 0.33	186 ± 2	88.42 ± 0.25	198 ± 1	90.39 ± 0.19	196 ± 0	90.57 ± 0.23
	✓	185 ± 3	89.08 ± 0.05	192 ± 2	89.17 ± 0.10	196 ± 1	91.01 ± 0.20	198 ± 1	91.12 ± 0.35
✓	×	194 ± 0	88.29 ± 0.25	193 ± 2	88.66 ± 0.31	190 ± 3	90.72 ± 0.28	193 ± 1	90.99 ± 0.40
	✓	198 ± 1	89.22 ± 0.25	196 ± 1	89.62 ± 0.03	199 ± 1	91.07 ± 0.13	200 ± 0	$\mathbf{92.38} \pm 0.17$

Table 9 The accuracy(%), MACs(K), BOPs(G) variance for Table 3 in the manuscript acquired by running experiments for 5 times.

Backbone	MACs	BOPs	Acc.
VGG-small	465 ± 10	7.37 ± 0.03	92.95 ± 0.10
	332 ± 12	3.85 ± 0.10	92.69 ± 0.26
	231 ± 9	2.29 ± 0.07	92.54 ± 0.18
ResNet20	38 ± 1	0.63 ± 0.02	92.58 ± 0.35
	31 ± 2	0.36 ± 0.02	92.35 ± 0.24
	37 ± 2	0.20 ± 0	92.38 ± 0.17

Table 10 The top-1 accuracy(%), MACs(G), BOPs(G) variance for Table 4 in the manuscript acquired by running experiments for 5 times.

Backbone	MACs	BOPs	Top-1
MobileNet-V2	0.27 ± 0.05	10.93 ± 0.05	71.22 ± 0.27
	0.22 ± 0	7.67 ± 0.05	70.78 ± 0.07
	0.20 ± 0.01	4.90 ± 0.02	70.38 ± 0.19
ResNet18	1.35 ± 0.08	55.98 ± 0.99	69.65 ± 0.19
	1.25 ± 0.06	31.28 ± 0.79	69.15 ± 0.14
	0.68 ± 0.07	7.40 ± 0.39	67.52 ± 0.27
ResNet50	3.25 ± 0.17	92.38 ± 0.21	76.50 ± 0.30
	3.03 ± 0.07	60.06 ± 0.20	76.28 ± 0.28
	2.10 ± 0.10	31.19 ± 0.52	75.95 ± 0.12

Table 11 The mAP(%), MACs(G), BOPs(G) variance for Table 5 in the manuscript acquired by running experiments for 5 times.

Backbone	MACs	BOPs	mAP
VGG16	18.08 ± 0.23	772.12 ± 5.33	69.5 ± 0.2
	14.93 ± 0.45	629.78 ± 10.33	68.7 ± 0.3
	14.01 ± 0.69	440.15 ± 3.78	66.9 ± 0.3
ResNet18	19.00 ± 0.15	542.17 ± 7.90	72.9 ± 0.2
	14.89 ± 0.22	338.13 ± 6.89	72.1 ± 0.2
	13.03 ± 0.46	295.29 ± 12.23	71.2 ± 0.7

volutional layer in the state space and supplemented an extra action a_k^p to sample the pruning ratio of compression strategy in the action space. The accuracy of the compressed model applied in the reward function was obtained via our performance predictor. Other implementation details were the same as those in Wang et al. (2019a). For evolutionary algorithms, we followed the same implementation details in Wang et al. (2020a) to search the optimal compression policy except that we deleted the network architecture components for each candidate. The accuracy of each candidate applied in the fitness function was acquired via our performance predictor. We imposed the resource constraint by limiting the BOPs of the compressed models during the search process.

E. Performance Variance of SeerNet

In order to show the performance variance of our SeerNet, we run SeerNet for 5 times including compression policy search and backbone training for results in Table 1-6. We report the mean and standard deviation of accuracies and computational complexity due to the variation, while the search cost and training cost are almost the same for each time. We leveraged the same complexity budget as that in Table 1-6 of the manuscript, and Table 7-12 show the experimental results with mean and standard deviation.

Table 12 The mAP(%), MACs(G), BOPs(G) variance for Table 6 in the manuscript acquired by running experiments for 5 times.

Backbone	MACs	BOPs	mAP
VGG16	20.92 ± 0.86	797.16 ± 8.27	22.6 ± 0.1
	19.52 ± 0.59	558.18 ± 7.10	22.2 ± 0.3
	15.59 ± 0.19	425.10 ± 2.23	21.3 ± 0
ResNet18	19.99 ± 0.15	482.35 ± 16.72	26.8 ± 0.1
	14.29 ± 0.11	318.90 ± 8.72	25.3 ± 0.3
	12.69 ± 0.39	289.75 ± 4.50	25.1 ± 0

Table 13 The MACs(G), BOPs(G), top-1 classification accuracy and search cost on ImageNet with pruning-only methods in MobileNet-V2. The search cost is demonstrated by GPU hours, where N means the number of deployment scenarios. The number in the bracket of search cost for mixed-precision quantization demonstrates the break-even point of baseline methods whose search cost is higher than our SeerNet.

Methods	MACs	BOPs	Comp.	Top-1	Cost
Baseline	0.33	337.9	–	71.72	–
AMC	0.23	236.5	1.43	70.90	$62.3N(13)$
NetAdapt	0.22	225.3	1.50	70.80	$95.6N(8)$
MetaPruning	0.22	222.2	1.52	71.20	$900{+}0.31N(1)$
SeerNet	0.22	227.6	1.48	71.52	$750{+}0.002N$

Table 14 The MACs(G), BOPs(G), top-1 classification accuracy and search cost on ImageNet with quantization-only methods in MobileNet-V2, ResNet18 and ResNet50.

Backbone	Methods	MACs	BOPs	Top-1	Cost
MobileNet-V2	Baseline	0.33	337.9	71.72	–
	HAQ	0.33	8.25	69.45	$51.1N(15)$
	SeerNet	0.33	8.08	70.64	$750{+}0.003N$
ResNet18	Baseline	1.81	1853.4	69.74	–
	HAWQ	1.81	34.00	68.45	$22.7N(23)$
	SeerNet	1.81	33.16	69.38	$500{+}0.002N$
ResNet50	Baseline	3.86	3952.6	76.40	–
	HAWQ	3.86	61.29	75.48	$34.5N(28)$
	SeerNet	3.86	58.49	76.25	$950{+}0.003N$

F. Performance of Pruning-only and Quantization-only Strategies

In this section, we evaluate SeerNet in the experimental settings with pruning-only and quantization-only strategies, where the backbone networks are only compressed by pruning and quantization policies in the

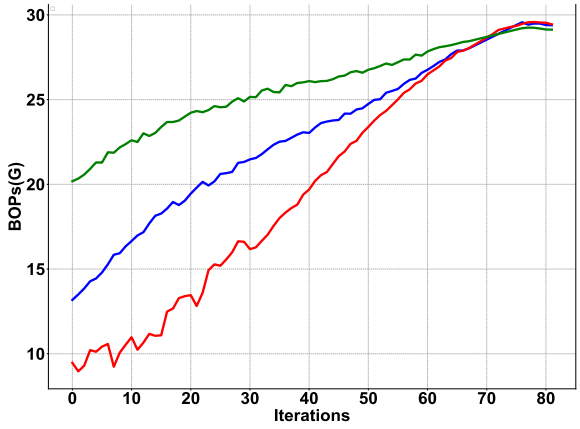


Fig. 8 Several examples w.r.t. the compression policy complexity during the optimization with different policy initializations, where no optimization paths stop early because of exceeding the computational cost budget. Different colors represent various initializations, where the BOPs constraint is 30G.

above settings respectively. We implemented SeerNet following the details introduced in Section 4.1 of the manuscript except for the modifications that the compression policy sampling for performance predictor training only contains pruning or quantization for the two settings. The compared methods include AMC (He et al., 2018b), NetAdapt (Yang et al., 2018), MetaPruning (Liu et al., 2019) for pruning-only methods and contain HAQ (Wang et al., 2019a) and HAWQ (Dong et al., 2019) for quantization-only methods. For pruning-only strategies, since only one shared experimental setting exists in AMC, NetAdapt and MetaPruning which employs MobileNet-V2 with 0.22G MACs for evaluation, we also assign the similar complexity constraint for fair comparison. For quantization-only strategies, we compare SeerNet with HAQ and HAWQ with MobileNet-V2, ResNet18 and ResNet50. Table 13 and 14 illustrate the results for pruning-only and quantization-only strategies respectively, where our SeerNet still outperforms the baseline methods by a sizable margin with much less marginal search cost.

G. Visualization of Policy Optimization

The presented barrier complexity loss in the compression policy optimization is amplified significantly for model complexity approaching the cost budget, which strictly limits the acquired pruning and quantization strategies within the complexity constraint. The complexity of the pruning and quantization policy in the optimization path usually keeps a margin from the computational cost budget. In order to empirically demonstrate the effectiveness of the barrier complexity loss, we implemented compression policy search for 50 times with different initialization, and the compression policies with complexity higher than the budget were never observed during the compression policy optimization. Figure 8 shows several examples w.r.t. the compression policy complexity during the optimization with different policy initializations, where no optimization paths stop early because of exceeding the computational cost budget.

Table 15 The BOPs(G), top-1 classification accuracy and search cost on ImageNet with random search baseline and our SeerNet in ResNet18.

Methods	W/A	MACs	BOPs	Comp.	Top-1	Cost
ResNet18						
Baseline	32/32	1.81	1853.4	–	69.74	–
RS	mixed	1.35	53.51	34.64	66.83	$62.1N(9)$
SeerNet	mixed	1.37	56.94	32.55	69.72	$500{+}0.003N$
RS	mixed	1.18	28.75	54.51	67.00	$62.1N(9)$
SeerNet	mixed	1.22	31.84	58.21	69.48	$500{+}0.004N$
RS	mixed	0.69	6.89	267.00	65.72	$62.1N(9)$
SeerNet	mixed	0.70	7.19	257.71	67.84	$500{+}0.004N$

H. Comparison with The Random Selection Baseline Method

To show the effectiveness of our search method, we conducted experiments to compare our SeerNet with random selection baseline method (RS) with ResNet18 on ImageNet. The pipeline of RS is demonstrated as follows: (a) randomly sampling k compression policies that satisfy the BOPs constraint, (b) exhaustively evaluating the acquired lightweight architectures, (c) selecting the one with the highest accuracy. With the same complexity constraint in Table 4 of the manuscript, we randomly sample 5 compression strategies that satisfy the BOPs budget for each random selection. Table 15 demonstrates the results, where the BOPs of RS is far from the budget and underperforms our SeerNet by a large margin regarding the accuracy.

References

Ehsan Abbasnejad, Damien Teney, Amin Parvaneh, Javen Shi, and Anton van den Hengel. Counterfactual vision and language learning. In *CVPR*, pages 10044–10054, 2020.

Maria-Florina Balcan, Andrei Broder, and Tong Zhang. Margin based active learning. In *COLT*, pages 35–50, 2007.

Sean Bell, C Lawrence Zitnick, Kavita Bala, and Ross Girshick. Inside-outside net: Detecting objects in context with skip pooling and recurrent neural networks. In *CVPR*, pages 2874–2883, 2016.

William H Beluch, Tim Genewein, Andreas Nürnberger, and Jan M Köhler. The power of ensembles for active learning in image classification. In *CVPR*, pages 9368–9377, 2018.

Joseph Bethge, Christian Bartz, Haojin Yang, Ying Chen, and Christoph Meinel. Meliusnet: Can binary neural networks achieve mobilenet-level accuracy? *arXiv preprint arXiv:2001.05936*, 2020.

Adrian Bulat and Georgios Tzimiropoulos. Bit-mixer: Mixed-precision networks with runtime bit-width selection. In *ICCV*, pages 5188–5197, 2021.

Han Cai, Chuang Gan, Tianzhe Wang, Zhekai Zhang, and Song Han. Once-for-all: Train one network and specialize it for efficient deployment. *arXiv preprint arXiv:1908.09791*, 2019.

Zhaowei Cai and Nuno Vasconcelos. Rethinking differentiable search for mixed-precision neural networks. In *CVPR*, pages 2349–2358, 2020.

Guobin Chen, Wongun Choi, Xiang Yu, Tony Han, and Manmohan Chandraker. Learning efficient object detection models with knowledge distillation. In *NIPS*, pages 742–751, 2017.

Jungwook Choi, Zhuo Wang, Swagath Venkataramani, Pierce I-Jen Chuang, Vijayalakshmi Srinivasan, and Kailash Gopalakrishnan. Pact: Parameterized clipping activation for quantized neural networks. *arXiv preprint arXiv:1805.06085*, 2018.

Xiaoliang Dai, Peizhao Zhang, Bichen Wu, Hongxu Yin, Fei Sun, Yanghan Wang, Marat Dukhan, Yunqing Hu, Yiming Wu, Yangqing Jia, et al. Chamnet: Towards efficient network design through platform-aware model adaptation. In *CVPR*, pages 11398–11407, 2019.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *CVPR*, pages 248–255, 2009.

Misha Denil, Babak Shakibi, Laurent Dinh, Nando De Freitas, et al. Predicting parameters in deep learning. In *NIPS*, pages 2148–2156, 2013.

Yinpeng Dong, Fangzhou Liao, Tianyu Pang, Hang Su, Jun Zhu, Xiaolin Hu, and Jianguo Li. Boosting adversarial attacks with momentum. In *CVPR*, pages 9185–9193, 2018.

Zhen Dong, Zhewei Yao, Amir Gholami, Michael W Mahoney, and Kurt Keutzer. Hawq: Hessian aware quantization of neural networks with mixed-precision. In *ICCV*, pages 293–302, 2019.

Włodzisław Duch and Jerzy Korczak. Optimization and global minimization methods suitable for neural networks. *Neural computing surveys*, 2:163–212, 1998.

Venice Erin Liong, Jiwen Lu, Gang Wang, Pierre Moulin, and Jie Zhou. Deep hashing for compact binary codes learning. In *CVPR*, pages 2475–2483, 2015.

Steven K Esser, Jeffrey L McKinstry, Deepika Bablani, Rathinakumar Appuswamy, and Dharmendra S Modha. Learned step size quantization. *arXiv preprint arXiv:1902.08153*, 2019.

Mark Everingham, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The pascal visual object classes (voc) challenge. *IJCV*, 88(2):303–338, 2010.

Christoph Feichtenhofer, Haoqi Fan, Jitendra Malik, and Kaiming He. Slowfast networks for video recognition. In *ICCV*, pages 6202–6211, 2019.

Chris Finlay, Aram-Alexandre Pooladian, and Adam Oberman. The logbarrier adversarial attack: making effective use of decision boundary information. In *ICCV*, pages 4862–4870, 2019.

Yarin Gal, Riashat Islam, and Zoubin Ghahramani. Deep bayesian active learning with image data. *arXiv preprint arXiv:1703.02910*, 2017.

Ruihao Gong, Xianglong Liu, Shenghu Jiang, Tianxiang Li, Peng Hu, Jiazhen Lin, Fengwei Yu, and Junjie Yan. Differentiable soft quantization: Bridging full-precision and low-bit neural networks. *arXiv preprint arXiv:1908.05033*, 2019.

Yash Goyal, Ziyan Wu, Jan Ernst, Dhruv Batra, Devi Parikh, and Stefan Lee. Counterfactual visual explanations. *arXiv preprint arXiv:1904.07451*, 2019.

Hai Victor Habi, Roy H Jennings, and Arnon Netzer. Hmq: Hardware friendly mixed precision quantization block for cnns. *arXiv preprint arXiv:2007.09952*, 2020.

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*, 2015a.

Song Han, Jeff Pool, John Tran, and William Dally. Learning both weights and connections for efficient neural network. In *NIPS*, pages 1135–1143, 2015b.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, pages 770–778, 2016.

Yang He, Guoliang Kang, Xuanyi Dong, Yanwei Fu, and Yi Yang. Soft filter pruning for accelerating deep convolutional neural networks. *arXiv preprint arXiv:1808.06866*, 2018a.

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

Yihui He, Ji Lin, Zhijian Liu, Hanrui Wang, Li-Jia Li, and Song Han. Amc: Automl for model compression and acceleration on mobile devices. In *ECCV*, pages 784–800, 2018b.

Andrew G Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*, 2017.

Itay Hubara, Matthieu Courbariaux, Daniel Soudry, Ran El-Yaniv, and Yoshua Bengio. Binarized neural networks. In *NIPS*, pages 4107–4115, 2016.

Qing Jin, Linjie Yang, and Zhenyu Liao. Adabits: Neural network quantization with adaptive bit-widths. In *CVPR*, pages 2146–2156, 2020.

Ajay J Joshi, Fatih Porikli, and Nikolaos Papanikolopoulos. Multi-class active learning for image classification. In *CVPR*, pages 2372–2379, 2009.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Alex Krizhevsky and Geoffrey Hinton. Learning multiple layers of features from tiny images. 2009.

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

Jiashi Li, Qi Qi, Jingyu Wang, Ce Ge, Yujian Li, Zhangzhang Yue, and Haifeng Sun. Oicsr: Out-in-channel sparsity regularization for compact deep neural networks. In *CVPR*, pages 7046–7055, 2019a.

Rundong Li, Yan Wang, Feng Liang, Hongwei Qin, Junjie Yan, and Rui Fan. Fully quantized network for object detection. In *CVPR*, pages 2810–2819, 2019b.

Xin Li and Yuhong Guo. Multi-level adaptive active learning for scene classification. In *ECCV*, pages 234–249, 2014.

Yawei Li, Shuhang Gu, Christoph Mayer, Luc Van Gool, and Radu Timofte. Group sparsity: The hinge between filter

pruning and decomposition for network compression. In *CVPR*, pages 8018–8027, 2020a.

Yuhang Li, Xin Dong, and Wei Wang. Additive powers-of-two quantization: A non-uniform discretization for neural networks. *ICLR*, 2020b.

Timothy P Lillicrap, Jonathan J Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. Continuous control with deep reinforcement learning. *arXiv preprint arXiv:1509.02971*, 2015.

Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *ECCV*, pages 740–755, 2014.

Baoyuan Liu, Min Wang, Hassan Foroosh, Marshall Tappen, and Marianna Pensky. Sparse convolutional neural networks. In *CVPR*, pages 806–814, 2015.

Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C Berg. Ssd: Single shot multibox detector. In *ECCV*, pages 21–37, 2016.

Zechun Liu, Baoyuan Wu, Wenhan Luo, Xin Yang, Wei Liu, and Kwang-Ting Cheng. Bi-real net: Enhancing the performance of 1-bit cnns with improved representational capability and advanced training algorithm. In *ECCV*, pages 722–737, 2018a.

Zechun Liu, Haoyuan Mu, Xiangyu Zhang, Zichao Guo, Xin Yang, Kwang-Ting Cheng, and Jian Sun. Metapruning: Meta learning for automatic neural network channel pruning. In *ICCV*, pages 3296–3305, 2019.

Zhuang Liu, Mingjie Sun, Tinghui Zhou, Gao Huang, and Trevor Darrell. Rethinking the value of network pruning. In *ICLR*, 2018b.

Qian Lou, Feng Guo, Minje Kim, Lantao Liu, and Lei Jiang. Autoq: Automated kernel-wise neural network quantization. In *ICLR*, 2019.

Christos Louizos, Max Welling, and Diederik P Kingma. Learning sparse neural networks through l_0 regularization. *arXiv preprint arXiv:1712.01312*, 2017.

Christos Louizos, Matthias Reisser, Tijmen Blankevoort, Efstratios Gavves, and Max Welling. Relaxed quantization for discretized neural networks. *arXiv preprint arXiv:1810.01875*, 2018.

Wenjie Luo, Alex Schwing, and Raquel Urtasun. Latent structured active learning. *NIPS*, 26:728–736, 2013.

Prem Melville and Raymond J Mooney. Diverse ensembles for active learning. In *ICML*, page 74, 2004.

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

Pavlo Molchanov, Arun Mallya, Stephen Tyree, Iuri Frosio, and Jan Kautz. Importance estimation for neural network pruning. In *CVPR*, pages 11264–11272, 2019.

Hanyu Peng, Jiaxiang Wu, Shifeng Chen, and Junzhou Huang. Collaborative channel pruning for deep networks. In *ICML*, pages 5113–5122, 2019.

Hai Phan, Dang Huynh, Yihui He, Marios Savvides, and Zhiqiang Shen. Mobinet: A mobile binary network for image classification. *arXiv preprint arXiv:1907.12629*, 2019.

Zhongnan Qu, Zimu Zhou, Yun Cheng, and Lothar Thiele. Adaptive loss-aware quantization for multi-bit networks. In *CVPR*, pages 7988–7997, 2020.

Mohammad Rastegari, Vicente Ordonez, Joseph Redmon, and Ali Farhadi. Xnor-net: Imagenet classification using binary convolutional neural networks. In *ECCV*, pages 525–542, 2016.

Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *NIPS*, pages 91–99, 2015.

Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *CVPR*, pages 4510–4520, 2018.

Burr Settles and Mark Craven. An analysis of active learning strategies for sequence labeling tasks. In *EMNLP*, pages 1070–1079, 2008.

Yawar Siddiqui, Julien Valentin, and Matthias Nießner. Viewal: Active learning with viewpoint entropy for semantic segmentation. In *CVPR*, pages 9433–9443, 2020.

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

Ilya Sutskever, James Martens, George Dahl, and Geoffrey Hinton. On the importance of initialization and momentum in deep learning. In *ICML*, pages 1139–1147, 2013.

Stefan Uhlich, Lukas Mauch, Kazuki Yoshiyama, Fabien Cardinaux, Javier Alonso Garcia, Stephen Tiedemann, Thomas Kemp, and Akira Nakamura. Differentiable quantization of deep neural networks. *arXiv preprint arXiv:1905.11452*, 2019.

Deepak Vasisht, Andreas Damianou, Manik Varma, and Ashish Kapoor. Active learning for sparse bayesian multilabel classification. In *KDD*, pages 472–481, 2014.

Sudheendra Vijayanarasimhan and Kristen Grauman. Large-scale live active learning: Training object detectors with crawled data and crowds. *IJCV*, 108(1-2):97–114, 2014.

Kuan Wang, Zhijian Liu, Yujun Lin, Ji Lin, and Song Han. Haq: Hardware-aware automated quantization with mixed precision. In *CVPR*, pages 8612–8620, 2019a.

Tianzhe Wang, Kuan Wang, Han Cai, Ji Lin, Zhijian Liu, Hanrui Wang, Yujun Lin, and Song Han. Apq: Joint search for network architecture, pruning and quantization policy. In *CVPR*, pages 2078–2087, 2020a.

Wenguan Wang, Hongmei Song, Shuyang Zhao, Jianbing Shen, Sanyuan Zhao, Steven CH Hoi, and Haibin Ling. Learning unsupervised video object segmentation through visual attention. In *CVPR*, pages 3064–3074, 2019b.

Ying Wang, Yadong Lu, and Tijmen Blankevoort. Differentiable joint pruning and quantization for hardware efficiency. In *ECCV*, pages 259–277, 2020b.

Ziwei Wang, Quan Zheng, Jiwen Lu, and Jie Zhou. Deep hashing with active pairwise supervision. In *ECCV*, pages 522–538, 2020c.

Ziwei Wang, Jiwen Lu, and Jie Zhou. Learning channel-wise interactions for binary convolutional neural networks. *TPAMI*, 43(10):3432–3445, 2021a.

Ziwei Wang, Han Xiao, Jiwen Lu, and Jie Zhou. Generalizable mixed-precision quantization via attribution rank preservation. In *ICCV*, pages 5291–5300, 2021b.

Ziwei Wang, Jiwen Lu, Ziyi Wu, and Jie Zhou. Learning efficient binarized object detectors with information compression. *TPAMI*, 44(6):3082–3095, 2022a.

Ziwei Wang, Changyuan Wang, Xiuwei Xu, Jie Zhou, and Jiwen Lu. Quantformer: Learning extremely low-precision vision transformers. *TPAMI*, pages 1–14, 2022b. doi: 10.1109/TPAMI.2022.3229313.

Wei Wen, Hanxiao Liu, Yiran Chen, Hai Li, Gabriel Bender, and Pieter-Jan Kindermans. Neural predictor for neural architecture search. In *ECCV*, pages 660–676, 2020.

Zhenyu Wu, Ziwei Wang, Zibu Wei, Yi Wei, and Haibin Yan. Smart explorer: Recognizing objects in dense clutter via interactive exploration. In *IROS*, pages 6600–6607, 2022.

Tien-Ju Yang, Andrew Howard, Bo Chen, Xiao Zhang, Alec Go, Mark Sandler, Vivienne Sze, and Hartwig Adam. Netadapt: Platform-aware neural network adaptation for mobile applications. In *ECCV*, pages 285–300, 2018.

Haibao Yu, Qi Han, Jianbo Li, Jianping Shi, Guangliang Cheng, and Bin Fan. Search what you want: Barrier panelty nas for mixed precision quantization. *arXiv preprint arXiv:2007.10026*, 2020.

Dongqing Zhang, Jiaolong Yang, Dongqiangzi Ye, and Gang Hua. Lq-nets: Learned quantization for highly accurate and compact deep neural networks. In *ECCV*, pages 365–382, 2018.