Boost Supernet Training with Contrastive Learning and Convergence-Considering Sampling Method

Yang Zhang[1], Meixi Liu[1], He Wei[1], Zhen Hou[1], Yangyang Tang[1], Haiyang Wu[1], Yuekui Yang[1,2]
[1]Machine Learning Platform Department, Tencent TEG [2]Department of Computer Science and Technology,
Beijing National Research Center for Information Science and Technology, Tsinghua University
{yizhizhang,meixiliu,whywei,yatang,scotthou,gavinwu,yuekuiyang}@tencent.com

Abstract

Instead of training each architecture independently in the search space, One-Shot NAS wraps all of them in one supernet which only needs to be trained once. However, the performance of the network sampled from the supernet can be inconsistent with the performance from which trained independently. To address this problem, we design a new framework by introducing a two-stage training procedure with a self-supervised pretrain stage to warm up the parameters of the super-network. We also propose a novel sampling method to improve the subnetwork selection by taking the convergence speed into consideration. A sampling factor is used to evaluate the convergence speed of the subnetworks. Our approach has been shown to be effective in experiments and eventually achieved 1st place in the Supernet Track of CVPR NAS Competition 2022.

1. Introduction

As machine learning techniques become the game changers in many problem settings for both industry and academy, Automatic Machine Learning (AutoML), which helps machine learning models learn and perform in a more effective way, has become an important research topic. Neural Architecture Search (NAS) is a popular approach that automatically finds the optimal model architecture for a particular task. NAS has been shown to be able to produce network architectures with better performance compared to manually designed ones in a dozen of computer vision and natural language processing tasks. However, typical NAS methods lead to a massive increase in network size and computation overhead [1].

Due to the inefficiency and computation cost of NAS, several different designs are proposed to boost the NAS learning. One-Shot NAS defines a supernet as the search space, which allows us to represent a wide variety of model architectures by this single model. During evaluation for model selection, it lets sub-networks simply inherits parameter weights from the trained supernet [1] [2]. Compared to independently training each model from scratch, One-Shot NAS potently reduces computational resources needed to learn large amount of different candidate models.

Accordingly, recent One-Shot based NAS researches manage to alleviate the ranking inconsistency of weights-inherited networks and ones which trained independently. Once-For-All [3] utilizes knowledge distillation to optimize the distances of evaluation accuracy between networks generated by supernet training and stand-alone training. It also incorporates with a so-called progressive shrinking algorithm, training networks in a particular order. Chu. et al [4] explores the impact of network sampling method, and develops a fair sampling procedure for subnet training.

In this paper, we illustrate a multi-stage supernet training routine, aiming to improve the ranking consistency of subnet architectures generated in two different ways, *i.e.*, supernet weight sharing and stand-alone training. We apply a self-supervised contrastive learning program to supernet training, enhancing the ability of network to fully extract image features. The network trained by contrastive learning will sooner be used in the supervised learning stage. By analyzing the classification performance of different subnet architectures, we find that the classification accuracy of various subnet is within a minute range of values, which makes maintaining ranking consistency a more difficult task. Therefore, based on observations of experiments, we replace the naive random sampling method with a convergence considering sampling method, which takes into account the number of layer parameters and the depth of layers. It is proven that our novel sampling method is able to address the problem described above.[1].

Besides, there is no need to do any network parameter cloning in our method. Compared to other One-Shot NAS solutions [5], it saves memory resource considerably.

[1] open-source code of our project: https://github.com/FuxiAutoML/CVPR_2022_NAS_Competition_Track1_1st

2. Method

2.1. Problem Definition

ResNet48 model, as the supernet in our case, is composed of a "stem" stage and 4 convolutional stages. Each stage contains several convolutional blocks. The stem stage contains one single convolutional block, the 1st, 2nd and 4th stage contain 5 blocks, and the 3rd stage contains 8 such blocks. Each convolutional block is made up by two convolutional layer and corresponding batch norm layers [6].

Search Space. For each convolutional layer, we assign a specific channel width expand ratio ranging from 0.7 to 1.0, the actual channel width is the base channel width of the current stage multiplied by the expand ratio, rounded to the closest multiple of 8. Also note that the total number of convolutional layers of a subnet is non-deterministic, a layer from supernet may or may not be a part of the subnet architecture. The search space can be formulated as below:

- Base channel widths for the stem and 4 stages: [64, 64, 128, 256, 512]
- Channel expand ratios: [0.7, 0.75, 0.8, 0.85, 0.9, 0.95, 1.0]
- Possible range of convolutional blocks for 4 stages: [2, 5], [2, 5], [2, 8], [2, 5]

The number of all network architectures is 5.69e+39.

Evaluation Metric. Our goal is to get the image classification accuracy for a variety of subnets on the IMAGENET dataset. The ultimate metric to evaluate our solution is the Kendall-Tau ranking correlation of the prediction accuracy between stand-alone trained subnets versus subnets drawn from the supernet model. Architectures of these subnets are sampled from a predefined set containing 45k networks.

2.2. Self-Supervised Contrastive Learning

To enhance the ability of modeling image features, we apply a self-supervised contrastive learning style to our supernet. Following [7], we regard the Contrastive Learning as a procedure of building dynamic dictionary, in which "keys" are encoder output of an image data.

ResNet Encoder Network. The encoder model is based on the original ResNet48, of which the linear classification layer is replaced by a Multi-Layer Perceptron called "neck" network. This neck network consists of two fully connected layers, the output is $e_{image} \in R^d$, which represents the d-dimension final embedding of the original image after feature extractions.

The dynamic dictionary is implemented as a queue data structure, and the size of dictionary(*i.e.* length of the queue) is decoupled with the batch size. For each step of mini-batch training, we maintain the queue dynamically by pushing the encoded representation of current mini-batch into the queue and popping out the oldest record out.

At the beginning, both two encoders share same parameter weights loaded from a pretrained ResNet model checkpoint. The neck is initialized by He initialization [8].

Contrastive Loss. We utilize a contrastive loss as the objective function. We denote the encoded query as q, and several encoded sampled keys stored in the dynamic dictionary as $k_1, k_2, ...$. We also denote the key in the dictionary which matches query q as k_+.

In practice, q and k_+ are representations of one image (applied with different random transformations) encoded by two different encoders. During training, the contrastive loss value will be minimized as q and k_+ become more similar while the difference between q and other unrelated k become larger. In summary, the contrastive loss can be formulated as:

$$L_{con} = -\log \frac{\exp\left(q \cdot k_+ / \tau\right)}{\sum_i \exp\left(q \cdot k_i / \tau\right)} \tag{1}$$

this loss can be viewed as a log loss of a softmax-based classifier trying to classify q to k_+. In the literature, it's commonly named as the InfoNCE loss. Note that τ in the formula above is called "temperature" coefficient. A smaller τ (usually less than 1.0) indicates more dramatic difference between positive class and negative ones. In our settings, τ is set in a range of [0.2, 0.02].

Momentum Update. By denoting the parameters of encoder f_k and f_q as θ_k and θ_q, we will update θ_k by the following formula:

$$\theta_k = \theta_q \cdot (1 - \lambda) + \theta_k \cdot \lambda \tag{2}$$

, λ is the so-called momentum coefficient. Empirically, λ is set to a value close to 1.0(*e.g.* λ = 0.999). In that case, the encoder f_k will evolve slowly and smoothly while encoder f_q is changing rapidly by gradient backpropagation.

With the momentum update mechanism, parameters of encoder network for dictionary keys are evolving in a fairly stable fashion, thus, negative samples fetched from the queue are more consistent during training.

After the 1st training stage described above, the ResNet backbone parameters of student network will be used as the beginning status of ResNet in the 2nd stage, while the neck MLP parameters shall be dropped. We still use the pretrained ResNet as the teacher network for knowledge distillation in the 2nd stage.

2.3. Convergence Considering Fair Sampling

While training supernet, a small number of sub-networks are sampled to be trained in each step. A fair sampling is proposed in [4], which allows all subnets to be trained the same number of times, regardless of the sampling order. However, the search space in [4] is uniformly distributed, and there is no overlap of parameters between these subnets. Therefore, a new strategy was proposed in [3]: train

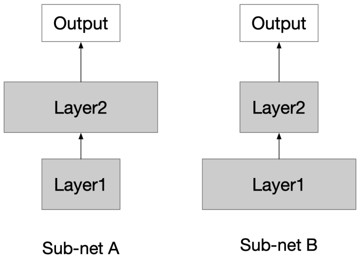

Figure 1. Parameter number of net A is equal to net B, but B is harder to converge, because gradients in Layer1 is easier to diffuse than Layer2 and B has more parameters in Layer1.

the larger networks first to protect the performance of small networks. However, we found that the scheme in [3] leads to a serious underestimation of large networks in our case.

Intuitively, large networks require more training samples compared to small networks. Parameters of the small network are part of the large network, and if the number of training samples is the same, the parameters of the small network will be updated by more times and converge faster. Meanwhile, larger networks demand more training steps to reach a satisfying fitting status.

Sampling Method. The convergence speed of sub-network is related to the total number of parameters. So it's natural to sample more subnets with larger parameters by calculating the amount of network parameters. What's more, the layers far away from the model output layer may suffer from the gradient vanishing (illustrated in Figure 1). We need to consider both the depth and the parameter amount of layers.

Accordingly, we introduce a new sample method called Convergence Considering Fair Sampling (CCFS), which utilizes a sampling factor to evaluate the convergence speed of each sub-network:

$$\Phi = \sum_i \alpha_i \cdot p_i \tag{3}$$

For a particular subnet architecture, p_i denotes the parameter amount of i-th layer α_i is a hyperparameter which describe the influence of the depth of the layer, α_i is higher when layer i is more far away from output layer, and Φ is the final sampling factor for the subnet. The sampling algorithm is detailed in Algorithm 1.

Algorithm 1: Convergence Considering Fair Sampling

```
Data: search space $S$, epochs $N$, final loss $L$ in
      Stage 2, data loader $D$, sample number $K$
initialize parameters in supernet;
for $i \leftarrow 1$ to $N$ do
    for data, label in $D$ do
        uniformly sampling $p$ subnetworks from $S$;
        calculate $\Phi_j$ for each of $p$ subnetworks;
        sort $\Phi$s and subnetworks as list $P$;
        $P_K \leftarrow Top_K(P, K)$;
        for $k \leftarrow 1$ to $K$ do
            calculate $L$ for subnetwork $k$ of $P_K$;
            accumulate gradients based on $L$;
        end
        update parameters by gradients;
    end
end
```

2.4. Knowledge Distillation

Integrated with the Convergence Considering Fair Sampling, we also exploit a knowledge distilling method in the 2nd stage. A widely used approach to transfer the knowledge from teacher model to student model is to force the student model outputs to be similar to the teacher's. Consider the logits output by student and teacher model, we have the probabilities that the input belongs to the classes given by student and teacher model respectively:

$$p(z_{i,s}) = \frac{\exp{(z_{i,s}/\tau)}}{\sum_j \exp{(z_{j,s}/\tau)}} \tag{4}$$

where z_i is the logit for the i-th class, $p(z_{i,s})$ is probabilities generated by student model. Note that probability $p(z_i, t)$ generated by teacher model is also named *Soft Targets*.

We can define the classification loss as the cross entropy loss L_{CE} between the probabilities given by student model and the ground truth label. The distillation loss is the distance between student model outputs and soft targets:

$$L_{Dis} = f_{Dis}(p(z_{i,s}), p(z_{i,t}))) \tag{5}$$

In our case, loss function f_{Dis} is set as cross entropy with soft labels. And the final loss function L in the 2nd training stage (used in Algorithm 1) is the sum of L_{CE} and L_{Dis}.

3. Experiments

Hyperparameter Settings. To evaluate our method, we train the supernet following the schema described above with some hyperparameter settings. We firstly train the supernet for stage 1 with batch size 256 and a momentum gradient optimizer whose learning rate set as 0.0008 and momentum as 0.9. The learning rate also follows a cosine decay strategy. The weight decay is set to 1e-4, which regularizes weights during training. In stage 2, batch size is kept same while learning rate is set to 0.0005. The cosine decay strategy and weight decay regularization is also applied here in stage 2. As for sampling factors in CCFS, α_i for stem and 4 stages are set as 1.0, 1.0, 0.55, 0.43 and 0.24.

Method	Ranking Correlation
vanilla	0.849
contrastive	**0.858**

Table 1. ranking correlation of vanilla and contrastive method.

Note that layers in same stage share same value of α_i in formula (3).

Data Augmentation. We apply a data augmentation process for each image fed into encoders. In practice, we employ transformations including random image resizing and horizontal flipping.

Experiment Metric. In our experiments, we evaluate the ranking consistency (Kendall-Tau metric) of the trained model. Note that there is a high relativity between Pearson correlation and the Kendall-Tau metric used as the final metric to determine which solution is better among all teams.

3.1. Results of Self-Supervised Learning

Table 1 shows the comparison between performances of supernet trained from scratch and supernet trained with an extra beginning contrastive learning stage. The "*vanilla*" denotes the supernet trained from scratch, i.e., only trained by the supervised classification task and knowledge distillation method in stage 2. The "*contrastive*" denotes the supernet trained by two stages illustrated in Sec 2.2. We can find that the final metric of *contrastive* outperforms the *vanilla*.

To further exhibit the self-supervised learning, we also tried some novel ideas for transferring network trained in Stage 1 to Stage 2, such as freezing parameters trained in Stage 1 and training only FC layer in Stage 2. However, the metric degrades dramatically. We believe that the number of parameters in FC layer is too small to play a vital role in learning the potential trends of different subnets.

3.2. Results of Sampling Methods

The comparison between traditional sampling methods and the CCFS (Convergence Considering Fair Sampling) is shown in table 2. It demonstrates that CCFS achieves a great improvement compared to other sampling methods.

With other settings kept same, CCFS gains an increment of 3.9% for the ranking correlation compared with FairNAS. Here *CCFS-M* takes only parameter number p_i into consideration to calculate Φ for subnets, while *CCFS-F* also considers gradient vanishing by setting various value for α_i.

4. Conclusion

In this paper, a novel compound supernet training procedure is proposed to improve the ranking correlation of evaluation metrics. The supernet training procedure is divided into two stages, namely, self-supervised training and traditional supervised training. In the first stage, inspired by the concept of contrastive learning, we reformat the original network to an image encoder, and facilitate the learning of vision features. In the second stage, we propose a convergence-considering method and further employ it to sample subnets with a novel distribution. Eventually, experiments demonstrate that our method significantly improves supernet capability and achieves better ranking consistency between supernet weight inheritance and stand-alone subnet learning.

Sampling Method	Ranking Correlation
FairNAS	0.819
OFA	0.823
CCFS-M	0.841
CCFS-F	**0.858**

Table 2. ranking correlation of different subnet sampling method.

References

[1] Youngkee Kim, Won Yun, Youn Lee, Soyi Jung, and Joongheon Kim. Trends in neural architecture search: Towards the acceleration of search. 08 2021. 1

[2] Gabriel Bender, Pieter-Jan Kindermans, Barret Zoph, Vijay Vasudevan, and Quoc V. Le. Understanding and simplifying one-shot architecture search. In *ICML*, 2018. 1

[3] Han Cai, Chuang Gan, and Song Han. Once for all: Train one network and specialize it for efficient deployment. *ArXiv*, abs/1908.09791, 2020. 1, 2, 3

[4] Xiangxiang Chu, Bo Zhang, Ruijun Xu, and Jixiang Li. Fairnas: Rethinking evaluation fairness of weight sharing neural architecture search. *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021. 1, 2

[5] Xiu Su, Shan You, Mingkai Zheng, Fei Wang, Chen Qian, Changshui Zhang, and Chang Xu. K-shot nas: Learnable weight-sharing for nas with k-shot supernets. *ArXiv*, abs/2106.06442, 2021. 1

[6] Kaiming He, X. Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 2

[7] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross B. Girshick. Momentum contrast for unsupervised visual representation learning. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020. 2

[8] Kaiming He, X. Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. *2015 IEEE International Conference on Computer Vision (ICCV)*, 2015. 2