Pi-NAS: Improving Neural Architecture Search by Reducing Supernet Training Consistency Shift

Jiefeng Peng[1,2]*,   Jiqi Zhang[1]*,   Changlin Li[3],   Guangrun Wang[4]†,
Xiaodan Liang[1],   Liang Lin[1]

[1]Sun Yat-sen University   [2]DarkMatter AI Research
[3]GORSE Lab, Dept. of DSAI, Monash University   [4]University of Oxford

{jiefengpeng,wanggrun,xdliang328}@gmail.com,
zhangjq49@mail2.sysu.edu.cn, changlin.li@monash.edu, linliang@ieee.org

Abstract

*Recently proposed neural architecture search (NAS) methods co-train billions of architectures in a supernet and estimate their potential accuracy using the network weights detached from the supernet. However, the ranking correlation between the architectures' predicted accuracy and their actual capability is incorrect, which causes the existing NAS methods' dilemma. We attribute this ranking correlation problem to the supernet training consistency shift, including **feature shift** and **parameter shift**. Feature shift is identified as dynamic input distributions of a hidden layer due to random path sampling. The input distribution dynamic affects the loss descent and finally affects architecture ranking. Parameter shift is identified as contradictory parameter updates for a shared layer lay in different paths in different training steps. The rapidly-changing parameter could not preserve architecture ranking. We address these two shifts simultaneously using a nontrivial supernet-Π model, called Π-NAS. Specifically, we employ a supernet-Π model that contains cross-path learning to reduce the feature consistency shift between different paths. Meanwhile, we adopt a novel nontrivial mean teacher containing negative samples to overcome parameter shift and model collision. Furthermore, our Π-NAS runs in an unsupervised manner, which can search for more transferable architectures. Extensive experiments on ImageNet and a wide range of downstream tasks (e.g., COCO 2017, ADE20K, and Cityscapes) demonstrate the effectiveness and universality of our Π-NAS compared to supervised NAS. See Codes[1].*
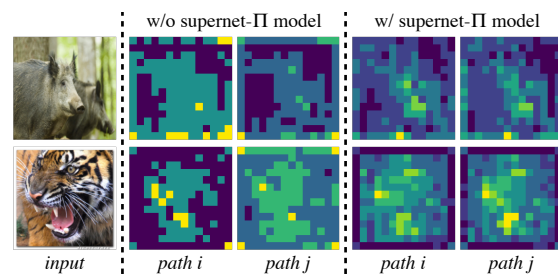
1. Introduction

Automatic neural architecture search (NAS) has been an intense longing in machine learning in the past four years.

(a) Feature shift. **Left:** Without the supernet-Π model, there is a feature shift between different paths' feature maps. **Right:** With supernet-Π model, the feature shift is significantly reduced.
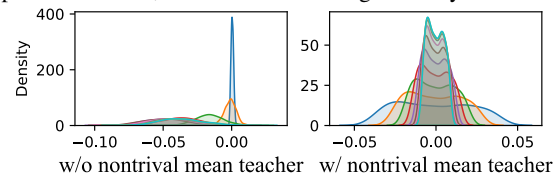
(b) Parameter shift. Different colors represent the distribution of parameters in different iterations. **Left:** without our nontrivial mean teacher, the parameter has significantly varying distributions in training. **Right:** with our nontrivial mean teacher, the parameter shift is significantly reduced.

Figure 1: Illustration of supernet training consistency shift.

Early works use reinforcement learning [69] or evolutionary algorithms [40] to discover high-performance architectures in the search space. The searching procedure usually costs thousands of GPU days for large datasets, as each sampled architecture needs training from scratch. Recently, to alleviate the heavy burden, weight sharing NAS methods [16, 9, 20, 54, 33, 30, 15, 36, 17, 1, 8] are widely used, where candidate architectures share weights and train simultaneously in a supernet[2]. After training, a candidate subnet's weights detached from the supernet are used to predict its actual performance. Despite the remarkable progress in efficiency, weight-sharing NAS's effectiveness

---

*Jiefeng Peng and Jiqi Zhang are co-first authors and share equal contributions. Their names are listed in alphabetical order.

†Corresponding Author.

[1]Code: https://github.com/Ernie1/Pi-NAS

---

[2]A supernet is an over-parameterized network that integrates the entire search space. Each architecture within the search space corresponds to a supernet's sub-net capturing the required operations.

is still unstable, i.e., it has a low ranking correlation between candidates' actual accuracies and accuracies estimated in supernet. In short, inaccurate architecture ranking is an inevitably critical problem in today's NAS.

In this paper, we attribute the ranking correlation problem to the supernet training consistency shift, including *feature shift* and *parameter shift*. **Feature shift** is identified as dynamic input distributions of a hidden layer. Specifically, a given layer's input feature maps always have an uncertain distribution due to random path sampling (see Figure 1a, left). This distribution uncertainty can hurt the architecture ranking correlation. Precisely, we can use the loss to measure the architecture accuracy, and we can link the accuracy ascent to gradient descent. Based on the back-propagation rule, a stable input distribution can guarantee a good ranking correlation. In contrast, the input distribution dynamic affects the loss descent and finally affects architecture ranking. **Parameter shift** is identified as contradictory parameter updates for a given layer. In supernet training, a given layer will always be present in different paths from iteration to iteration (see Figure 1b, left). The parameter in this layer may have a contradictory update from iteration to iteration. These unstable updates lead to varying parameters' distributions, hurting the architecture ranking correlation in two ways. On the one hand, stable parameters can ensure a correct loss descent and guarantee an accurate architecture ranking, while frequent parameter change could not preserve architecture ranking. On the other hand, varying parameters can also result in a feature shift, further hurting architecture ranking correlation. In summary, both feature shift and parameter shift can hurt the architecture ranking correlation. **Detailed experimental analysis** in Section 4 provide solid evidence to support this analysis.

Motivated by consistency regularization methods [29, 44], we propose a nontrivial supernet-Π model, called Π-**NAS**, to reduce these two shifts simultaneously. Specifically, to cope with the *feature shift*, we propose a novel supernet-Π model. We evaluate each data point through two randomly sampled paths, then apply a consistency cost between the two predictions to penalize the feature consistency shift between different paths. As shown in Figure 1a (right), our method can significantly reduce the feature shit and thus can improve the architecture ranking correlation. To address the *parameter shift*, we propose a novel nontrivial mean teacher model by maintaining an exponential moving average of weights in supernet teacher. Although a mean teacher can stabilize the parameters in single network training, it could be trapped in a trivial solution and lead to a model collision in supernet training. Our nontrivial mean teacher novelly contains appropriate negative samples to avoid such a model collision. An impressive result of our method in reducing the parameter shift is shown in Figure 1b (right). In brief, our Π-NAS can reduce the supernet

training consistency shift and thus improve the architecture ranking, which is critical for NAS's effectiveness.

One by-product that could not be ignored is that our Π-NAS runs in an unsupervised manner, which has an additional gain that existing supervised NAS methods do not have. Concretely, similar to unsupervised representation learning that can learn general features, our Π-NAS can search for more transferrable and universal architectures than supervised NAS counterparts.

Since the "good architectures" in previous NAS search spaces usually have considerable computation complexity, using these search spaces for evaluation lacks interpretability. To evaluate our Π-NAS, we design a nontrivial search space based on 16-layer ResNet-50. Our searched models on this space achieve a state-of-the-art top-1 accuracy of 81.6% on ImageNet, surpassing ResNeSt-50 by 0.5% with comparable computation cost. We also validate Π-NAS on NAS-Bench-201 with CIFAR-10, beating state-of-the-art NAS methods and verifying our method's effectiveness. In addition, our Π-NAS models keep state-of-the-art on many downstream tasks (e.g., COCO 2017 detection and segmentation, ADE20K segmentation, and Cityscapes segmentation), demonstrating the universality of our Π-NAS.

Overall, this paper makes three contributions.

- We attribute the inaccurate architecture ranking to the supernet training consistency shift, including feature and parameter shifts. Then we provide a detailed empirical analysis of how these two shifts are making NAS methods ineffective.

- We propose a Π-NAS method with two key components, i.e., a supernet-Π model and a nontrivial mean teacher, to address feature shift and parameter shift, respectively. Notably, our nontrivial mean teacher model introduces appropriate negative samples to avoid being trapped in a trivial solution.

- Our Π-NAS method shares the merit of unsupervised representation learning, i.e., the universality property. We can search for architectures that are more transferrable and universal than supervised NAS methods. Substantial empirical results are obtained on ImageNet and a wide range of downstream tasks to demonstrate the effectiveness and universality of our Π-NAS.

2. Related Work

Neural Architecture Search (NAS). NAS has attracted increasing research attention in recent years. Early NAS works [67, 14, 37, 69, 3, 40, 42] consume a huge amount of computation resources to train thousands of candidate models from scratch while using an agent (an RNN controller or evolution algorithm) to explore better-performing architectures in the search space. To alleviate the computational overhead caused by the training process, researchers starts to share the weights among candidate archi-

tectures [16, 9, 20, 54, 33, 30, 15, 36, 17, 1, 8]. Gradient-based weight sharing methods [36, 9, 54, 63] jointly optimize the shared network parameters and the architecture choosing factors by gradient descent. In one-shot methods [20, 16, 8, 4, 30], the supernet is first optimized with path sampling, and then sub-models are sampled and evaluated with the weights inherited from the supernet. Despite the acceleration of weight sharing, these approaches still suffer a critical issue on their effectiveness [4, 16, 33]. Existing attempting on solving this issue includes ensuring optimization fairness among all child models [16], reducing the search space greedily during training [33], modularizing the large search space into blocks using an intermediate knowledge distillation [30] and constraining the subnet optimization to prevent multi-model forgetting [64, 65]. Recently, unsupervised NAS methods are also starting to attract research interest [35, 58, 31, 66, 48].

Reducing Consistency Shift. *Feature shift* is represented as the instability of the network to the perturbation of an input image. Penalizing the consistency shift can help develop the network's tolerance to incorrect labels and improve the classification accuracy in semi-supervised learning [2, 41, 29, 44, 60, 56, 38, 7, 57, 52, 50, 47]. [29] proposes Π-model to encourage consistent output for input with different augmentation and dropout, and extend the Π-model by temporal ensembling the network's output for each input, to retain the consistency of the outputs. *Parameter shift* is represented as the instability of network parameters. To address the parameter shift, a mean teacher model [44] refines the temporal ensembling by averaging the model weights rather than outputs, which has also been used to stabilize weight sharing training [32]. In this paper, we attribute NAS's inefficiency to incorrect architecture ranking caused by supernet training consistency shift, i.e., feature shift and parameter shift. Since Π model is a classical tool to reduce feature shift, we propose a supernet-Π model to address the feature shift. Our supernet-Π model is a novel one as we use a novel formulation of cross-path learning. On the other hand, mean-teacher is widely adopted to reduce parameter shift because it can reliably reduce the implausible uncertainties. Hence, we introduce mean teacher to address our parameter shift. Although a mean teacher can be employed to stabilize the parameters in single network training, it could be trapped in a trivial solution and lead to a model collision in supernet training. Our nontrivial mean teacher novelly contains appropriate negative samples to avoid such a model collision. In summary, our method is a nontrivial NAS method aiming at closing the supernet training consistency shift, but not a straightforward combination of the NAS and Π model and mean teacher.

Contrastive Learning. Recent contrastive learning-based methods have brought a leap in unsupervised representa-tion learning [39, 55, 26, 45, 68, 22, 11, 51]. Being cast as either the dictionary look-up task [55, 22] or the consistent learning task [45, 11], these methods learn discriminative representations by bringing the representation of different views of the same image closer and spreading representations of views from different images apart. MoCo [22, 13] uses an exponential moving average (EMA) encoder to generate predictions and keep a large bank of the historical predictions as the negative samples. In BYOL [19], the online network with a predictor is trained to be consistent with the EMA target network without requiring negative pairs. However, straightforward applying the technique from contrastive learning to NAS could be either unnecessary or unsuccessful. Due to the training consistency shift, there will be a feature shift in a pair of samples in contrastive learning, especially in negative sample pairs. This makes the supernet optimization unstable and hard to convergent. In contrast, our Π-NAS contains a cross-path training formulation that can satisfactorily address the feature shift problem.

3. Methodology

We first briefly introduce the dilemma of NAS, i.e., inaccurate architecture ranking, then attribute incorrect architecture ranking to the supernet training consistency shift, including feature shift and parameter shift. Then, we propose a nontrivial supernet-Π model with two key components, i.e., a supernet-Π model and a nontrivial mean teacher, to address feature shift and parameter shift, respectively. At last, we search promising architecture in linear evaluation.

3.1. Dilemma of NAS

Inaccurate architecture ranking. Let $\mathcal{A}$ denote the architecture search space. $\alpha \in \mathcal{A}$ and ω_α are the network architecture and the network weights, respectively. As mentioned above, NAS aims to find an optimal pair $(\alpha^*, \omega_\alpha^*)$ such that the model performance is maximized in search space $\mathcal{A}$. The searching procedure can be formulated as two subproblems. The first one is a *architecture training* that trains the network weights of given architectures. The second one is an *architecture search* that searches for an architecture with the best performance if trained. As training each architecture from scratch to convergence is prohibitive in practice due to the high computation cost, recently, weight-sharing NAS was proposed. [9, 20, 54, 33, 30] propose to train for different candidates concurrently via a weight sharing strategy, encoding the search space $\mathcal{A}$ in an over-parameterized supernet. Thus, all candidate architectures can inherit their weights immediately from the supernet. However, the proxy weights borrowed from the supernet do not adequately indicate network weights trained from scratch to convergence, as each subgraph is not fairly and sufficiently optimized in supernet. This may lead to a low ranking correlation between the candidates' predicted accuracy and their actual capability, which causes the inef-
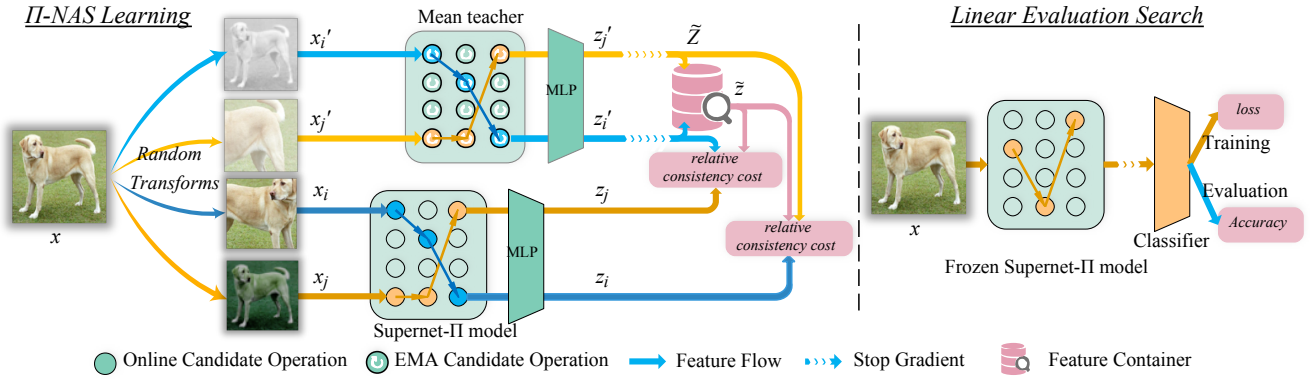
Figure 2: An overview of proposed Π-NAS method. In Π-*NAS learning*, the input image is transformed into four different views that are then separately routed through the supernet-Π model and nontrivial mean teacher to calculate relative consistency cost with negative targets. The outputs of the mean teacher are saved in the container to serve as negative targets in the future. In *Linear Evaluation Search*, the classifier is trained analogous to linear evaluation, and the accuracy is used as the metrics of architecture search.

fectiveness of architecture search. We identify this as the dilemma of NAS.

Supernet training consistency shift. So, what causes the dilemma of NAS? In this paper, we attribute the inaccurate architecture ranking to the supernet training consistency shift, which contains feature shift and parameter shift.

Feature shift is identified as dynamic input distributions of a hidden layer. Let $\mathbf{x}_l$ denote the input of layer l and $\mathbf{y}_l$ denote its output. $\mathbf{w}_l$ is its network weights. Since the final architecture accuracy is inaccessible during the training, we use the loss $\mathcal{L}$ to measure the architecture accuracy, and the accuracy ascent can be connected to the loss descent. According the chain rule of differentiation in the back-propagation algorithm, we have: $\frac{\partial \mathcal{L}}{\partial \mathbf{w}_l} = \frac{\partial \mathcal{L}}{\partial \mathbf{y}_l}\frac{\partial \mathbf{y}_l}{\partial \mathbf{w}_l} = \frac{\partial \mathcal{L}}{\partial \mathbf{y}_l}\mathbf{x}_l$. This indicates architecture ranking-preserving is highly dependent on the inputs $\mathbf{x}_l$. But for a given layer l, due to random path sampling in supernet, the preceding path varies, and the input $\mathbf{x}_l$ also varies. We thus should guarantee a stable $\mathbf{x}_l$ to preserve a good architecture ranking correlation. Otherwise, an input distribution dynamic impacts the loss descent and finally affects architecture ranking.

Parameter shift is identified as contradictory parameter updates for a given layer. In supernet training, a given layer l will always be present in different paths from iteration to iteration. Its weights may have a contradictory update from iteration to iteration, i.e., $\mathbf{w}_l^{t+1} \leftarrow \mathbf{w}_l^t - \frac{\partial \mathcal{L}^t}{\partial \mathbf{w}_l^t}$. The rapidly-varying $\mathbf{w}_l$ will hurt the architecture ranking correlation in two ways. On the one hand, the loss descent is not only connected to $\frac{\partial \mathcal{L}}{\partial \mathbf{w}_l}$ but is also connect $\mathbf{w}_l - \frac{\partial \mathcal{L}}{\partial \mathbf{w}_l}$. This indicates that stable parameters can ensure a correct loss descent and guarantee an accurate architecture ranking, while frequently-varying parameters could not preserve architecture ranking. On the other hand, since the input $\mathbf{x}_l$ is generated by the network weights of the previous layers, varying parameters can also result in a feature shift, which further hurts architecture ranking correlation.

In summary, both feature shift and parameter shift can

hurt the architecture ranking correlation, further making NAS methods ineffective. **Detailed experimental analysis** in Section 4 provide evidence to support this analysis.

3.2. Π-NAS: A Nontrivial Supernet-Π Model

As discussed, reducing the supernet training consistency shift can alleviate the dilemma of NAS. In the following, we design a novel and effective nontrivial supernet-Π model, including a supernet-Π model and a nontrivial mean teacher model, to address feature shift and parameter shift, respectively. Our Π-NAS can successfully preserve the architecture ranking and thus improve NAS's effectiveness.

Supernet-Π model. To guarantee a stable input distribution, we are devoted to penalizing the inconsistency between the same input predictions through different sampled paths. Motivated by a Π model, we evaluate data point x through two randomly sampled paths, denoted as path i and j, to get its representations $\{z_i, z_j, z_i', z_j'\}$. Note that we obtain representations z and z' with different views of augmentation, i.e., $z = f(x)$ and $z' = f'(x)$, where f and f' are mapping functions of the supernet model. Without loss of generality, we define f and f' as the student/teacher models. Normally, the student and the teacher are identical.

After obtaining evaluations of the same input x, we define a cross-path consistency cost as follow:

$$\mathcal{L}_{Con} = -\mathop{\mathbb{E}}_{X}[\mathcal{D}(z_i, z_j') + \mathcal{D}(z_j, z_i')] \tag{1}$$

where X and $\mathcal{D}$ denote a training data set and a consistency metric, respectively. Figure 2 shows a pipeline of our supernet-Π model with cross-path learning. By minimizing Eqn. 1, one could reduce the feature consistency shift caused by different random paths and thus stabilize the distributions of input features of a hidden layer.

In brief, we formulate our method under the Π framework with cross-path learning, i.e., supernet-Π model. Extensive experiments show a remarkable improvement in the architecture ranking correlation.

Nontrivial mean teacher model. Besides addressing feature shift, we also intend to reduce parameter shift by smoothing parameter updates from iteration to iteration. Inspired by mean-teacher [44], we propose to maintain an exponential moving average weights for teacher model rather than barely replicate from student model in supernet-Π model training. Formally, we denote $\mathcal{W}_t$ as parameters of student mapping function f at training step t. Then, weights of mean teacher model f' can be defined as:

$$\mathcal{W}'_t = \lambda \mathcal{W}'_{t-1} + (1-\lambda)\mathcal{W}_t \qquad (2)$$

where $\lambda \in [0,1]$ is a smoothing coefficient hyper-parameter.

Although the capability of a mean teacher to stabilize the parameters is obvious, it could be trapped in a trivial solution in the supernet-Π model. Specifically, barely optimizing consistency loss might lead to model collapse. For example, representations that are constant across arbitrary inputs are always entirely consistent. To circumvent this problem, we introduce appropriate negative samples to our model, i.e., nontrivial mean teacher model. Formally, an additive consistency cost is:

$$\mathcal{L}_{Add} = \mathop{\mathbb{E}}_{X}\left[\mathop{\mathbb{E}}_{\widetilde{Z}}[\mathcal{D}(z_i,\widetilde{z}) + \mathcal{D}(z_j,\widetilde{z})]\right] \qquad (3)$$

where $\widetilde{Z}$ represents a whole collection of negative samples $\widetilde{z}$, and $\widetilde{z} \in \widetilde{Z}$. Note that negative samples $\widetilde{z}$ can be collected from our nontrivial mean teacher model by reusing the previous predictions (see the *Feature Container* in Figure 2). A relative consistency cost can be written as: $\mathcal{L}_{Ref} = \mathcal{L}_{Con} + \mathcal{L}_{Add}$.

Since our target is to maximize the consistency metric between positive samples while minimizing the negative ones, we can formulate the optimization as the categorical cross-entropy of classifying the positive samples, with $\frac{\exp(\mathcal{D}(z,z'))}{\sum_{\widetilde{z}}\exp(\mathcal{D}(z,\widetilde{z}))+\exp(\mathcal{D}(z,z'))}$ being the prediction. We model consistency metric $\mathcal{D}$ with dot-product similarity as $\mathcal{D}(z,z') = z^T z'$. Thus the final loss function of Π-NAS is formulated as:

$$\mathcal{L} = -\mathop{\mathbb{E}}_{X}\Big[\log\frac{e^{z_i^T z_j'}}{\sum_{\widetilde{Z}} e^{z_i^T \widetilde{z}} + e^{z_i^T z_j'}} + \log\frac{e^{z_j^T z_i'}}{\sum_{\widetilde{Z}} e^{z_j^T \widetilde{z}} + e^{z_j^T z_i'}}\Big]. \quad (4)$$

3.3. Linear Evaluation Search

After optimizing the nontrivial supernet-Π model with $\mathcal{W}$, an architecture search is conducted by evaluating the representation capability of candidates α. Inspired by the standard linear evaluation protocol [28, 21] using in self-supervised learning, we train a linear classifier on the top of the frozen representation, i.e., without updating the supernet parameters $\mathcal{W}$ nor the batch statistics. Specifically, the linear classifier Fc is also optimized via a common weight sharing strategy. Then, we estimate the capability of the sub-model by its accuracy $\mathcal{R}_{val}$ on the validation set and search for the best performance:

$$\alpha^* = \mathop{\arg\max}_{\alpha \in \mathcal{A}} \mathcal{R}_{val}(Fc(\mathcal{W}_\alpha, \alpha; X, Y)) \qquad (5)$$

where $\mathcal{W}_\alpha$ is the sub-architecture α's parameters inherited directly from parameters $\mathcal{W}$.

Thanks to Π-NAS learning and linear evaluation searching, our Π-NAS not only improves the search effectiveness but also shows the superiority in searching for more transferable and universal architectures. Finally, an overview of our Π-NAS is presented in Figure 2.

4. Experiments

4.1. Implementation Details

Search space and dataset. We construct our supernet based on 16-layer ResNet-50 by replacing the residual bottleneck in each layer with 4 candidate *Split-Attention* blocks [62] of radix s, cardinality x and width d. Thus our search space $\mathcal{A}$ includes 4^{16} architectures.

- **Block0**: $1s1x64d$
- **Block1**: $2s1x64d$
- **Block2**: $1s2x42d$
- **Block3**: $2s2x40d$

Note that **Block1** is the building block of ResNeSt-50 [62].

We deliberately design such search space by two considerations. **First**, these four candidate blocks have similar Params and FLOPs to avoid performance gain at the cost of model complexity since models with higher complexity often achieve higher accuracy. Thus, our search space is a nontrivial space to examine NAS's effectiveness. **Second**, our search space is similar with ResNet rather than the recent works [20, 54, 30, 36] since the experiments demonstrate that variants of ResNet are more efficient in practice even though the statics are in the opposite. As shown in Table 2, with the same top-1 accuracy on ImageNet, the latency of ResNeSt-50 surpasses EfficienNet-B3 [43] by a margin of 14.5% even though with $2.9\times$ more FLOPs. To further reduce the training consistency shift, we share the bottleneck's downsample operation among all candidate blocks in the same layer. The advantage of *downsample-sharing* strategy will be illustrated in Section 4.5.

Our Π-NAS is evaluated on ImageNet, a state-of-the-art classification dataset widely used in recent NAS methods [20, 54, 30]. For the search procedure, we randomly pick out 50 images per class from the original 1.28M training set to build a 50k validation set, and the reset of images is used as a training set for supernet learning. All of our ImageNet results are tested on the original validation set.

Training details. We perform our Π-NAS in 3 stages: Π-NAS learning, linear evaluation, and architecture search.

In Π-NAS learning, inspired by [12], we use an augmentation strategy of random resize&crop, color jitter, color drop, Gaussian blur, and horizontal flip. Besides, we employ a 2-layer MLP as the supernet head. The smoothing coefficient λ of the *mean teacher* in Eqn. (2) is set to 0.999 in practice. The relative consistency loss is optimized by an SGD optimizer with a learning rate of 0.03, a momentum of

Table 1: Image classification fast results on the validation set. (Acc@S: top-1 accuracy in supernet)

Model	Params	FLOPs	Acc@S	Acc@1	Acc@5
ResNeSt-50	27.5M	5.42G	64.6%	80.7%	95.3%
Π-NAS-α (ours)	27.1M	5.38G	65.0%	81.2%	95.4%
Π-NAS-β (ours)	27.2M	5.39G	65.1%	81.2%	95.6%
Π-NAS-γ (ours)	27.0M	5.30G	65.0%	81.1%	95.6%
Π-NAS-δ (ours)	26.9M	5.30G	65.0%	81.0%	95.4%
Π-NAS-ϵ (ours)	26.9M	5.42G	65.0%	81.0%	95.4%

0.9, and a weight decay of 10^{-4}. We adopt a cosine decay learning rate schedule to train for 100 epochs with a total batch size of 192 on 8 NVIDIA GTX 2080Ti GPUs.

As for linear evaluation, we fetch the optimized supernet-Π model and replace the 2-layer MLP with a random initialized 1000-dimensional linear classifier. Only the linear classifier is trained on ImageNet for 100 epochs while the supernet's parameters $\mathcal{W}$ are frozen. At each training step, the linear classifier's inputs are obtained across stochastic paths from the supernet. Note that the batch statistics are used instead of tracked statistics in batch normalization (BN) layers to avoid inaccurate statistics across different sampled paths. Only random resize&crop, horizontal flip are used for data augmentation. We train the classifier with a total batch size of 256 for 100 epochs using a cross-entropy loss and an SGD optimizer with an initial learning rate of 30, a momentum of 0.9, and a weight decay of 0. The learning rate decays by 0.1 at 60 and 80 epochs.

In architecture search, the candidate architectures are evaluated separately with the top-1 accuracy on the 50k Imagenet validation set mentioned above. Again, to avoid the inaccurate batch statistics in BN, we pick out a further 50k images from the rest of the training set to recalculate the statistics for each optional path. Then, we adopt a search algorithm, Action Space [53], to seek candidates with the best performance with a maximum sample size of 1000.

4.2. Experiments on ImageNet

Fast results of searched models. As shown in Table 1, we first evaluate the top 5 models searched by our Π-NAS as well as the ResNeSt-50 (*Block1*) in a fast training setting. All the models are trained from scratch on the original ImageNet training set for 270 epochs with PyTorch-Encoding [61] following the same setting of ResNeSt-50 except using a total batch size of 512 instead of 8192 due to the limit of GPU memory. Our models significantly outperform ResNeSt-50 by an average margin of 0.4%, even with fewer parameters and FLOPs. In particular, all the searched top models achieve similar top-1 accuracy in supernet and training from scratch, respectively, which proves the effectiveness of our Π-NAS from another side.

Comparison with the state-of-the-art models. We select one of the searched models Π-NAS-α as our best model,

Table 2: Image classification results on the validation set. *img/sec* and *GPU* denote the inference speed and the GPU memory usage of the model performing inference, respectively, with a batch size of 128 on one NVIDIA GTX 2080Ti GPU. † Using crop size 300, otherwise 224. ∗ Searching for a bunch of sub-optimal solutions with close accuracy.

Model	Params	FLOPs	img/sec	GPU	Accuracy
ResNet-50 [24]	25.6M	4.12G	835.9	2.55G	78.4
SENet-50 [27]	27.7M	4.25G	-	-	78.9
SKNet-50 [34]	27.5M	4.47G	-	-	79.2
EfficientNet-B3†[43]	12.2M	1.88G	490.5	9.25G	81.1
ResNeSt-50 [62]	27.5M	5.42G	561.6	4.16G	81.1
Searched Models on Our Search Space $\mathcal{A}$ from NAS Methods					
SPOS [20]	27.1M	5.43G	536.4	4.12G	81.04±0.03
FairNAS [16]	26.9M	5.31G	541.7	3.87G	81.05±0.06
DNA [30]	26.8M	5.41G	571.6	3.71G	81.1*
FBNetV2 [46]	26.8M	5.29G	478.7	3.89G	81.1*
TuNAS [5]	26.8M	5.39G	554.8	4.95G	81.1*
Π-NAS-*cls* (ours)	27.1M	5.38G	556.8	4.07G	**81.6**

denoted as Π-NAS-*cls*, on ImageNet classification, considering a trade-off between performance and efficiency. We retrain ResNet-50 [24] (always undertrained in previous NAS works), ResNeSt-50 and our searched models on ImageNet under the same settings with an augmentation scheme, named AugMix [25]. For a fair comparison with the state-of-the-art NAS methods, we apply them on our search space $\mathcal{A}$. For SPOS [20] and FairNAS [16], we manipulate the same architecture search procedure as ours. For DNA [30], we select the candidate block with the minimum loss in each layer to build as its top model. For FBNetV2 [46] and TuNAS [5], we treat our search space as four possible channel decisions in each layer to apply the channel masking scheme. As we can see in Table 2, Π-NAS-*cls* marks a new state-of-the-art top-1 accuracy 81.6%, surpassing ResNeSt-50 by a large margin of 0.5% in a similar computation complexity. By contrast, in our nontrivial search space, the previous NAS methods seem stuck at the local optima near ResNeSt-50, verifying the advantage of Π-NAS to reduce the supernet training consistency shift. Moreover, even though having more computation complexity, our Π-NAS-*cls* achieves higher performance than EfficientNet-B3 [43] with lower latency and less GPU memory in practice. Notably, the results in Table 2 suggest that our Π-NAS-*cls* not only achieves state-of-the-art performance but also runs at a fast speed indeed.

Model ranking. As discussed in Section 1, a strong ranking correlation between candidates' actual and predicted performance in the supernet is essential to the effectiveness of NAS. Here, we compare our ranking correlation with DNA [30] and SPOS [20]. We use the top 5 architectures in Table 1 and randomly sample other eight architectures from the search space and train them in a fast setting described above to obtain their top-1 accuracy training from

Table 3: Ranking correlations (in Kendall's Tau metric) of diverse NAS methods in our search space.

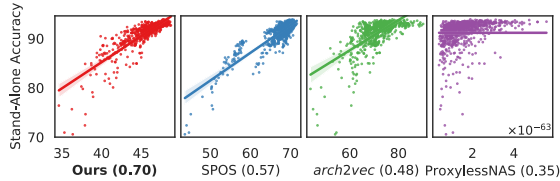| Method	Ours	DNA	SPOS	FairNAS	FBNetV2	TuNAS
Classification	**0.79**	0.45	0.19	0.36	0.32	0.14
Instance seg.	**0.51**	0.38	0.18	-	-	-

Figure 3: Ranking correlations on 792 architectures on NAS-Bench-201 [18] on CIFAR-10 without *skip connection* and *zero* operations compared to SPOS [20], *arch2vec* [59] and ProxylessNAS [9].

Table 4: Results on NAS-Bench-201 on CIFAR-10.

Method	Ours	SPOS	*arch2vec*	ProxylessNAS	WPL	GDAS-NSAS
Test(%)	**93.83±0**	93.57±0	92.53±0.32	92.08±0.03	90.92±0.11	93.55±0.16

Table 5: Instance segmentation results with Mask-RCNN [23] on the COCO 2017 validation set.

Model	AP^{Box}	AP^{Mask}
ResNet-50 [24]	39.93±0.04	35.99±0.06
ResNeSt-50 [62]	42.81±0.02	38.14±0.01
Π-NAS-*cls* (ours)	43.72	39.13
Π-NAS-*trans* (ours)	**44.11±0.04**	**39.48±0.02**

Table 6: Semantic segmentation results with DeeplabV3 [10] on the validation set of ADE20K and Cityscapes.

Model	ADE20K		Cityscapes
	pixAcc	mIoU	mIoU
ResNet-50 [24]	80.66±0.27	42.74±0.64	78.42±0.30
ResNeSt-50 [62]	81.22±0.05	45.18±0.06	80.08±0.20
Π-NAS-*trans* (ours)	**81.31±0.04**	**45.49±0.02**	**80.40±0.30**

scratch, then fetch their predicted performances in the supernet of each method to compute the ranking correlations. The second row of Table 3 suggests the advanced effectiveness of Π-NAS as it predicts the model's performance much more correctly. As analyzed in Section 3.1, this is due to the training consistency shift problem, which will be further discussed in Section 4.5.

4.3. Experimenet on NAS-Bench-201 Benchmarks

We additionally validate our Π-NAS on a popular cell-based search space, NAS-Bench-201 [18], on CIFAR-10 dataset. This search space is represented as a DAG, where each edge is associated to an operation with 5 options: *zero, skip connection, 1×1 convolution, 3×3 convolu-*
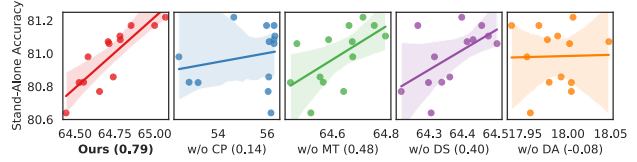

Figure 4: Ranking correlations of diverse NAS methods. Values within the parentheses are their corresponding Kendall's Tau. Our Π-NAS gains the best correlation, indicating that Π-NAS effectively reduces the training consistency shift.(CP: *cross-path learning*; MT: *mean teacher*; DS: *downsample-sharing*; DA: *learning different augmented views of the same image*)

tion and 3×3 *average pooling*. This DAG has 4 nodes, where each node represents the sum of feature maps transformed through the edges pointing to this node. For the sake of simplicity, though we train the supernet involving all 5 operations, we predict the performances of all 792 architectures without *zero* and *skip connection* operations to measure the ranking correlation to their ground-truth performances. As shown in Figure 3 and Table 4, our method significantly outperforms SPOS [20], *arch2vec* [59] (an unsupervised NAS method), ProxylessNAS [9] (a differentiable method), WPL [6] (a different solution to address parameter shift) and GDAS-NASA [65] by a clear margin, verifying our method's effectiveness and compatibility.

4.4. Experiments on Transfer Learning

Instance segmentation results. To explore the transferability of our Π-NAS models, we first evaluate them on a widely used transfer learning task, instance segmentation, which simultaneously solves the problem of object detection and semantic segmentation. We train the Mask-RCNN [23] on COCO-2017 with our searched models as its backbone following the instructions of [62, 49]. Rather than one model, we evaluate all the 13 architectures (used in 4.2 *Model ranking*) with pretrain models on ImageNet. Also, we study the ranking correlation by averaging the bounding box mAP (AP^{bb}) and mask mAP (AP^{mk}) as the actual performance. As shown in the third row of Table 3, the effectiveness of our Π-NAS stays superior, which indicates that our approach can search for architectures that are more transferrable and universal. Note that we choose the architecture with the best performance as a transferable model, Π-NAS-*trans* (*a.k.a.* Π-NAS-γ, one of our top 5 searched architectures), for the transfer learning. Table 5 shows that both of Π-NAS-*trans* and Π-NAS-*cls* outperform ResNeSt-50 by a significant margin (0.91% and 1.30% in AP^{bb}).

Semantic segmentation results. We further transfer Π-NAS-*trans* to the downstream task of semantic segmentation on ADE20K and Cityscapes datasets. We train DeeplabV3[10] with the implementation of PyTorch-
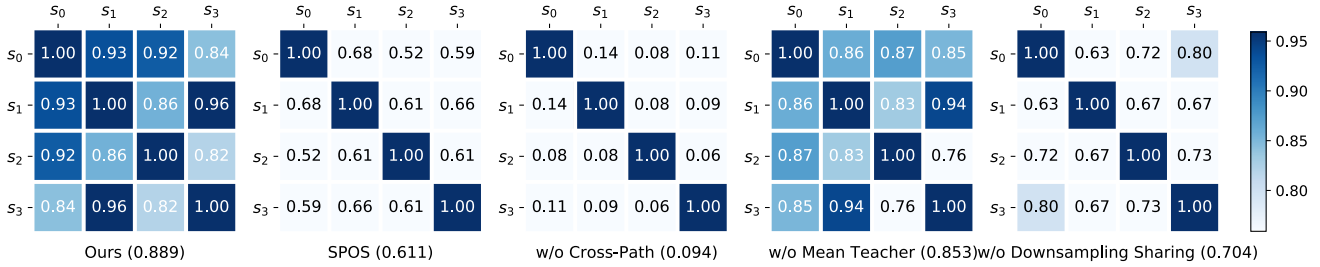
Figure 5: Cosine similarity matrices of the outputs from the last layer of 4 paths in each supernet variant. The value within the parentheses is the average after stripping out the diagonal. Our Π-NAS achieves the overall highest feature similarity, intuitively explaining its effectiveness to reduce the supernet's training consistency shift.

Encoding and the settings from[62]. For the ADE20K dataset, we train the model for 120 epochs with a base image size of 520 and cropped image size of 480. As for the Cityscapes dataset, the model is trained for 240 epochs; the base image size is 2048; the cropped image size is 768. We also follow [62] to use multi-scale evaluation with flipping. Results are shown in Table 6, both of which demonstrate the advantage of our Π-NAS-*trans*.

4.5. Ablation Study

Effectiveness of components. To evaluate the impact of our Π-NAS separately, we first distinguish it from SPOS by *cross-path learning*, *mean teacher* and *downsample-sharing*. As shown in Table 7 and Figure 4, we test the combination methods with Kendall's Tau as a ranking correlation between their models' predicted and actual performance. Adopting the same testing scheme in Section 4.2, we apply each method on their supernets' training and then evaluate the 13 architectures (used in 4.2 *Model ranking*). As we can see, the Supernet-Π model reduces by 0.31 without the mean-teacher, which indicates *mean teacher* plays a role in high ranking correlation. The most notable thing is that without *cross-path learning*, the method lost its effectiveness as SPOS. Obviously, *cross-path learning* is the essential component in our Π-NAS. *Downsample-sharing* also shows its strength in predicting accurate performance for candidate architectures with a 0.39 improvement. Note that when we try to perform Π-NAS without *nontrivial mean teacher*, the supernet converged quickly to a state that outputs all zeros, which disables the distinguishing ability of the model (see Table 7).

Feature consistency and ranking correlation. As analyzed in Section 3, *training consistency shift* damages the ranking correlation of NAS. To further demonstrate this statement, we explore and visualize the feature similarity from the last layer across paths. For example, we randomly sample 4 architectures except the last layer are *Block0*, *Block1*, *Block2* and *Block3* respectively, which are denoted as s_0, s_1, s_2 and s_3. Then we evaluate the feature cosine similarity between each pair of them. Figure 5 shows

Table 7: Effectiveness of each component of our Π-NAS. (CP: *cross-path learning*; MT: *mean teacher*; DS: *downsample-sharing*)

Method	CP	MT	DS	nontrivial	Kendall's Tau
SPOS [20]			✓		0.19
S-Π model	✓		✓	✓	0.48
Ours w/o CP		✓	✓	✓	0.14
Ours w/o DS	✓	✓		✓	0.40
Ours w/o nontrival	✓	✓	✓		collision
Ours	✓	✓	✓	✓	**0.79**

the embedding feature similarity of different methods. By correlating Figure 4, we found that a high feature consistency lead to a strong ranking correlation of supernet, which demonstrates convincingly our motivation. Notably, Figure 5 also proves our Π-NAS indeed reduces *supernet training consistency shift*, especially for *cross-path learning*.

5. Conclusion

This paper recognizes the importance of architecture ranking in NAS and attributes the ranking correlation problem to the supernet training consistency shift, including feature shift an parameter shift. To address these two shifts, we propose a nontrivial supernet-Π model, *i.e.*, Π-NAS. Specifically, we propose a supernet-Π model with cross-path learning to reduce feature shift and a nontrivial mean teacher to cope with parameter shift. Notably, our Π-NAS can search for more transferable and universal architectures than supervised NAS. Extensive experiments on many tasks demonstrate the search effectiveness and universality of our Π-NAS compared to the NAS counterparts.

References

[1] Youhei Akimoto, Shinichi Shirakawa, Nozomu Yoshinari, Kento Uchida, Shota Saito, and Kouhei Nishida. Adaptive stochastic natural gradient method for one-shot neural architecture search. In *ICML*, pages 171–180, 2019. 1, 3

[2] Philip Bachman, Ouais Alsharif, and Doina Precup. Learning with pseudo-ensembles. In *NeurIPS*, pages 3365–3373, 2014. 3

[3] Bowen Baker, Otkrist Gupta, Nikhil Naik, and Ramesh Raskar. Designing neural network architectures using reinforcement learning. *arXiv preprint arXiv:1611.02167*, 2016. 2

[4] Gabriel Bender, Pieter-Jan Kindermans, Barret Zoph, Vijay Vasudevan, and Quoc V. Le. Understanding and simplifying one-shot architecture search. In *ICML*, pages 549–558, 2018. 3

[5] Gabriel Bender, Hanxiao Liu, Bo Chen, Grace Chu, Shuyang Cheng, Pieter-Jan Kindermans, and Quoc V. Le. Can weight sharing outperform random architecture search? an investigation with tunas. In *CVPR*, June 2020. 6

[6] Yassine Benyahia, Kaicheng Yu, Kamil Bennani Smires, Martin Jaggi, Anthony C Davison, Mathieu Salzmann, and Claudiu Musat. Overcoming multi-model forgetting. In *International Conference on Machine Learning*, pages 594–603. PMLR, 2019. 7

[7] David Berthelot, Nicholas Carlini, Ian Goodfellow, Nicolas Papernot, Avital Oliver, and Colin A Raffel. Mixmatch: A holistic approach to semi-supervised learning. In *NeurIPS*, pages 5049–5059, 2019. 3

[8] Andrew Brock, Theodore Lim, James M. Ritchie, and Nick Weston. SMASH: one-shot model architecture search through hypernetworks. In *ICLR*, 2018. 1, 3

[9] Han Cai, Ligeng Zhu, and Song Han. ProxylessNAS: Direct neural architecture search on target task and hardware. In *ICLR*, 2019. 1, 3, 7, 12, 13

[10] Liang-Chieh Chen, G. Papandreou, Florian Schroff, and H. Adam. Rethinking atrous convolution for semantic image segmentation. *ArXiv*, abs/1706.05587, 2017. 7

[11] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. *arXiv preprint arXiv:2002.05709*, 2020. 3

[12] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey E. Hinton. A simple framework for contrastive learning of visual representations. *ArXiv*, abs/2002.05709, 2020. 5

[13] Xinlei Chen, Haoqi Fan, Ross Girshick, and Kaiming He. Improved baselines with momentum contrastive learning. *arXiv preprint arXiv:2003.04297*, 2020. 3

[14] Yukang Chen, Gaofeng Meng, Qian Zhang, Shiming Xiang, Chang Huang, Lisen Mu, and Xinggang Wang. RENAS: reinforced evolutionary neural architecture search. In *CVPR*, pages 4787–4796, 2019. 2

[15] Xiangxiang Chu, Bo Zhang, Jixiang Li, Qingyuan Li, and Ruijun Xu. Scarletnas: Bridging the gap between scalability and fairness in neural architecture search. *arXiv preprint arXiv:1908.06022*, 2019. 1, 3, 12

[16] Xiangxiang Chu, Bo Zhang, Ruijun Xu, and Jixiang Li. Fairnas: Rethinking evaluation fairness of weight sharing neural architecture search. *CoRR*, abs/1907.01845, 2019. 1, 3, 6

[17] Xuanyi Dong and Yi Yang. Searching for a robust neural architecture in four GPU hours. In *CVPR*, pages 1761–1770, 2019. 1, 3

[18] Xuanyi Dong and Yi Yang. Nas-bench-201: Extending the scope of reproducible neural architecture search. In *ICLR*, 2020. 7, 12, 13

[19] Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre H Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Avila Pires, Zhaohan Daniel Guo, Mohammad Gheshlaghi Azar, et al. Bootstrap your own latent: A new approach to self-supervised learning. *arXiv preprint arXiv:2006.07733*, 2020. 3

[20] Zichao Guo, Xiangyu Zhang, Haoyuan Mu, Wen Heng, Zechun Liu, Yichen Wei, and Jian Sun. Single path one-shot neural architecture search with uniform sampling. In *ECCV*. Springer, 2020. 1, 3, 5, 6, 7, 8, 12, 13

[21] Raia Hadsell, Sumit Chopra, and Yann LeCun. Dimensionality reduction by learning an invariant mapping. In *CVPR*. IEEE, 2006. 5

[22] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *CVPR*, 2020. 3

[23] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross B. Girshick. Mask r-cnn. *ICCV*, 2017. 7

[24] Kaiming He, X. Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *CVPR*, 2016. 6, 7

[25] Dan Hendrycks, Norman Mu, E. D. Cubuk, Barret Zoph, J. Gilmer, and Balaji Lakshminarayanan. Augmix: A simple data processing method to improve robustness and uncertainty. *ArXiv*, abs/1912.02781, 2020. 6

[26] R Devon Hjelm, Alex Fedorov, Samuel Lavoie-Marchildon, Karan Grewal, Phil Bachman, Adam Trischler, and Yoshua Bengio. Learning deep representations by mutual information estimation and maximization. *arXiv preprint arXiv:1808.06670*, 2018. 3

[27] Jie Hu, Li Shen, and Gang Sun. Squeeze-and-excitation networks. In *CVPR*, 2018. 6

[28] Alexander Kolesnikov, Xiaohua Zhai, and Lucas Beyer. Revisiting self-supervised visual representation learning. In *CVPR*, pages 1920–1929, 2019. 5

[29] Samuli Laine and Timo Aila. Temporal ensembling for semi-supervised learning. *arXiv preprint arXiv:1610.02242*, 2016. 2, 3

[30] Changlin Li, Jiefeng Peng, Liuchun Yuan, Guangrun Wang, Xiaodan Liang, Liang Lin, and Xiaojun Chang. Block-wisely supervised neural architecture search with knowledge distillation. In *CVPR*, 2020. 1, 3, 5, 6, 12

[31] Changlin Li, Tao Tang, Guangrun Wang, Jiefeng Peng, Bing Wang, Xiaodan Liang, and Xiaojun Chang. BossNAS: Exploring hybrid CNN-transformers with block-wisely self-supervised neural architecture search. In *ICCV*, 2021. 3

[32] Changlin Li, Guangrun Wang, Bing Wang, Xiaodan Liang, Zhihui Li, and Xiaojun Chang. Dynamic Slimmable Network. In *CVPR*, 2021. 3

[33] Xiang Li, Chen Lin, Chuming Li, Ming Sun, Wei Wu, Junjie Yan, and Wanli Ouyang. Improving one-shot nas by suppressing the posterior fading. In *CVPR*, 2020. 1, 3

[34] Xiang Li, Wenhai Wang, Xiaolin Hu, and Jian Yang. Selective kernel networks. In *CVPR*, 2019. 6

[35] Chenxi Liu, Piotr Dollár, Kaiming He, Ross Girshick, Alan Yuille, and Saining Xie. Are labels necessary for neural architecture search? *arXiv preprint arXiv:2003.12056*, 2020. 3

[36] Hanxiao Liu, Karen Simonyan, and Yiming Yang. DARTS: differentiable architecture search. In *ICLR*, 2019. 1, 3, 5

[37] Renato Negrinho and Geoffrey J. Gordon. Deeparchitect: Automatically designing and training deep architectures. *CoRR*, abs/1704.08792, 2017. 2

[38] Avital Oliver, Augustus Odena, Colin A Raffel, Ekin Dogus Cubuk, and Ian Goodfellow. Realistic evaluation of deep semi-supervised learning algorithms. In *NeurIPS*, pages 3235–3246, 2018. 3

[39] Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*, 2018. 3

[40] Esteban Real, Sherry Moore, Andrew Selle, Saurabh Saxena, Yutaka Leon Suematsu, Jie Tan, Quoc Le, and Alex Kurakin. Large-scale evolution of image classifiers. *arXiv preprint arXiv:1703.01041*, 2017. 1, 2

[41] Mehdi Sajjadi, Mehran Javanmardi, and Tolga Tasdizen. Regularization with stochastic transformations and perturbations for deep semi-supervised learning. In *NeurIPS*, pages 1163–1171, 2016. 3

[42] Mingxing Tan, Bo Chen, Ruoming Pang, Vijay Vasudevan, Mark Sandler, Andrew Howard, and Quoc V Le. Mnasnet: Platform-aware neural architecture search for mobile. In *CVPR*, pages 2820–2828, 2019. 2

[43] M. Tan and Quoc V. Le. Efficientnet: Rethinking model scaling for convolutional neural networks. *ArXiv*, abs/1905.11946, 2019. 5, 6

[44] Antti Tarvainen and Harri Valpola. Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results. In *NeurIPS*, 2017. 2, 3, 5

[45] Yonglong Tian, Dilip Krishnan, and Phillip Isola. Contrastive multiview coding. *arXiv preprint arXiv:1906.05849*, 2019. 3

[46] Alvin Wan, Xiaoliang Dai, Peizhao Zhang, Zijian He, Yuandong Tian, Saining Xie, Bichen Wu, Matthew Yu, Tao Xu, Kan Chen, Peter Vajda, and Joseph E. Gonzalez. Fbnetv2: Differentiable neural architecture search for spatial and channel dimensions. In *CVPR*, June 2020. 6

[47] Guangcong Wang, Jian-Huang Lai, Wenqi Liang, and Guangrun Wang. Smoothing adversarial domain attack and p-memory reconsolidation for cross-domain person re-identification. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020. 3

[48] Guangrun Wang, Liang Lin, Rongcong Chen, Guangcong Wang, and Jiqi Zhang. Joint learning of neural transfer and architecture adaptation for image recognition. *IEEE Transactions on Neural Networks and Learning Systems (T-NNLS)*, 2021. 3

[49] Guangrun Wang, Guangcong Wang, Keze Wang, Xiaodan Liang, and Liang Lin. Grammatically recognizing images with tree convolution. In Rajesh Gupta, Yan Liu, Jiliang Tang, and B. Aditya Prakash, editors, *KDD '20: The 26th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Virtual Event, CA, USA, August 23-27, 2020*, pages 903–912. ACM, 2020. 7

[50] Guangrun Wang, Guangcong Wang, Xujie Zhang, Jianhuang Lai, Zhengtao Yu, and Liang Lin. Weakly supervised person re-id: Differentiable graphical learning and a new benchmark. *IEEE Transactions on Neural Networks and Learning Systems*, 32(5):2142–2156, 2020. 3

[51] Guangrun Wang, Keze Wang, Guangcong Wang, Philip H. S. Torr, and Liang Lin. Solving inefficiency of self-supervised representation learning. 2021. 3

[52] Guangcong Wang, Xiaohua Xie, Jianhuang Lai, and Jiaxuan Zhuo. Deep growing learning. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2812–2820, 2017. 3

[53] Linnan Wang, Saining Xie, Teng Li, Rodrigo Fonseca, and Yuandong Tian. Sample-efficient neural architecture search by learning action space. *ArXiv*, abs/1906.06832, 2019. 6

[54] Bichen Wu, Xiaoliang Dai, Peizhao Zhang, Yanghan Wang, Fei Sun, Yiming Wu, Yuandong Tian, Peter Vajda, Yangqing Jia, and Kurt Keutzer. Fbnet: Hardware-aware efficient convnet design via differentiable neural architecture search. In *CVPR*, 2019. 1, 3, 5

[55] Zhirong Wu, Yuanjun Xiong, Stella X Yu, and Dahua Lin. Unsupervised feature learning via non-parametric instance discrimination. In *CVPR*, pages 3733–3742, 2018. 3

[56] Qizhe Xie, Zihang Dai, Eduard Hovy, Minh-Thang Luong, and Quoc V Le. Unsupervised data augmentation for consistency training. *arXiv preprint arXiv:1904.12848*, 2019. 3

[57] Qizhe Xie, Minh-Thang Luong, Eduard Hovy, and Quoc V Le. Self-training with noisy student improves imagenet classification. In *CVPR*, pages 10687–10698, 2020. 3

[58] Shen Yan, Yu Zheng, Wei Ao, Xiao Zeng, and Mi Zhang. Does unsupervised architecture representation learning help neural architecture search? *NeurIPS*, 33, 2020. 3

[59] Shen Yan, Yu Zheng, Wei Ao, Xiao Zeng, and Mi Zhang. Does unsupervised architecture representation learning help neural architecture search? In *NeurIPS*, 2020. 7, 12, 13

[60] Xiaohua Zhai, Avital Oliver, Alexander Kolesnikov, and Lucas Beyer. S4l: Self-supervised semi-supervised learning. In *ICCV*, pages 1476–1485, 2019. 3

[61] Hang Zhang. PyTorch-Encoding. https://github.com/zhanghang1989/PyTorch-Encoding, 2018. 6

[62] Hang Zhang, Chongruo Wu, Zhongyue Zhang, Yi Zhu, Zhi-Li Zhang, Haibin Lin, Yu e Sun, Tong He, Jonas Mueller, R. Manmatha, M. Li, and Alex Smola. Resnest: Split-attention networks. *ArXiv*, abs/2004.08955, 2020. 5, 6, 7, 8

[63] Miao Zhang, Huiqi Li, Shirui Pan, Xiaojun Chang, Zongyuan Ge, and Steven W. Su. Differentiable neural architecture search in equivalent space with exploration enhancement. In *NeurIPS*, 2020. 3

[64] Miao Zhang, Huiqi Li, Shirui Pan, Xiaojun Chang, and Steven Su. Overcoming multi-model forgetting in one-shot nas with diversity maximization. In *CVPR*, pages 7809–7818, 2020. 3

[65] Miao Zhang, Huiqi Li, Shirui Pan, Xiaojun Chang, Chuan Zhou, Zongyuan Ge, and Steven W Su. One-shot neural architecture search: Maximising diversity to overcome catastrophic forgetting. *IEEE Annals of the History of Computing*, 2020. 3, 7

[66] Xuanyang Zhang, Pengfei Hou, Xiangyu Zhang, and Jian Sun. Neural architecture search with random labels. In *CVPR*, 2021. 3

[67] Zhao Zhong, Junjie Yan, Wei Wu, Jing Shao, and Cheng-Lin Liu. Practical block-wise neural network architecture generation. In *CVPR*, pages 2423–2432, 2018. 2

[68] Chengxu Zhuang, Alex Lin Zhai, and Daniel Yamins. Local aggregation for unsupervised learning of visual embeddings. In *ICCV*, pages 6002–6012, 2019. 3

[69] Barret Zoph and Quoc V Le. Neural architecture search with reinforcement learning. *arXiv preprint arXiv:1611.01578*, 2016. 1, 2