Supplementary Materials for DisWOT: Student Architecture Search for Distillation WithOut Training

Peijie Dong[1†] Lujun Li[2†*] Zimian Wei[1†]
[1] National University of Defense Technology, [2] Chinese Academy of Sciences
[1]{dongpeijie, weizimian16}@nudt.edu.cn, [2]lilujunai@gmail.com

A. More Comparisons and Discussions

In this section, we provide more analysis and discussion about DisWOT from different aspects.

A.1. DisWOT under different teacher models.

When the teacher model becomes larger, the fixed hand-designed model would have huge teacher-student gaps, limiting the performance gain. DisWOT aims to solve this problem by searching the suitable student architecture for different teacher models. According to the results in Table 1, the accuracy of the student network ResNet20 [9] is unable to make consistent gains as the size of the teacher network increases. Our proposed DisWOT enables a consistent increase in student network performance as the teacher network capacity increases on search space S_0. In addition, DisWOT† achieves a performance gain of about 2% when stronger distillers are adopted.

Table 1. Top-1 accuracy (%) of ResNet20 with KD [10], student (DisWOT) with KD [10], student (DisWOT) with DisWOT† on search space S_0 under different teachers.

Teacher	ResNet20	DisWOT	DisWOT†
ResNet32	70.24	71.01	71.85
ResNet44	70.56	71.25	72.12
ResNet56	70.98	71.63	72.56
ResNet110	70.79	71.84	72.92

A.2. Ranking correlation metrics

We denote the ground-truth (GT) performance and approximated scores of architectures $\alpha_i (i = 1, ..., N)$ as $\beta_i (i = 1, ..., N)$ and $\gamma_i (i = 1, ..., N)$, respectively, and the ranking of the GT and estimated score β_i, γ_i as $r_i, k_i \in \{1, ..., N\}$. Three correlation criteria is adopted in this paper. Pearson coefficient (r), Kendall's Tau (τ), Spearman coefficient (ρ).

- Pearson correlation coefficient (Linear Correlation): $r = \text{corr}(\beta, \gamma)/\sqrt{\text{corr}(\beta, \beta)\text{corr}(\gamma, \gamma)}$.
- Kendall's Tau correlation coefficient: The relative difference of concordant pairs and discordant pairs $\tau = \sum_{i<j} \text{sgn}(\beta_i - \beta_j)\text{sgn}(\gamma_i - \gamma_j)/\binom{M}{2}$.
- Spearman correlation coefficient: The Pearson correlation coefficient between the ranking variables $\rho = \text{corr}(r, k)/\sqrt{\text{corr}(r, r)\text{corr}(k, k)}$.

Pearson measures the linear relationship between two variables, while Kendall's Tau and Spearman measure the monotonic relationship. They return a value between -1 and 1, with -1 indicating an inverse correlation, 1 indicating a positive correlation, and 0 representing no relationship.

In search space S_0, we evaluate and verify the ranking consistency for all the architectures in the search space. In search space S_1, we randomly sampled 50 sub-networks from the search space to calculate the ranking consistency results due to the excessive time overhead of the full measurement and repeated each experiment 10 times.

A.3. Detailed analysis about vanilla-distilling training disparity

In this paper, we notice an interesting and non-trivial observation: the discrepancy between the model's performance under vanilla training and under distillation training. We present more analysis in detail here from three aspects.

Ranking correlation degrade. In Table 2, we tabulate the ranking correlations for different zero-proxies with distilling and vanilla training on the search space S_0. The existing zero-proxies' distillation correlations are reduced by $5\% \sim 69\%$ than the vanilla correlation. This common issue shows that existing zero-shot NAS methods score sub-optimal student architectures for a given teacher model. DisWOT not only has a better correlation for distillation but also is free of vanilla-distill ranking consistency degradation.

Correlation visualization of different scores. Figure 1 demonstrates the ranking consistency of NWOT [21] for distillation accuracy and vanilla accuracy, and there is a

*Corresponding author, † equal contribution, PD conducted main experiments, LL proposed ideas and led the project & writing.

Table 2. Details experiments of the discrepancy between vanilla accuracy and distillation accuracy on search space S_0.

Method	Ranking with distill accuracy			Ranking with vanilla accuracy			Ranking gap for distill and vanilla accuracy		
	Kendall's Tau	Spearman	Pearson	Kendall's Tau	Spearman	Pearson	Kendall's Tau	Spearman	Pearson
FLOPs [1]	51.61	72.92	76.40	58.74	79.47	79.19	7.13 (↓)	6.55 (↓)	2.79 (↓)
Fisher [1]	62.86	81.37	20.90	81.68	95.28	70.24	18.82 (↓)	13.91 (↓)	49.34 (↓)
Grad_norm [1]	63.75	82.35	39.35	84.76	96.55	76.07	21.01 (↓)	14.2 (↓)	36.72 (↓)
NWOT [21]	31.87	45.66	48.99	40.29	56.46	56.23	8.42 (↓)	10.80 (↓)	7.24 (↓)
Plain [1]	10.72	13.57	-0.91	54.98	77.12	67.55	44.26 (↓)	63.55 (↓)	68.46 (↓)
SNIP [14]	67.22	85.07	51.09	84.66	96.38	77.83	17.44 (↓)	11.31 (↓)	26.74 (↓)
DisWOT	73.98	91.38	84.83	73.02	91.26	82.98	0.96 (↑)	0.12 (↑)	1.85 (↑)

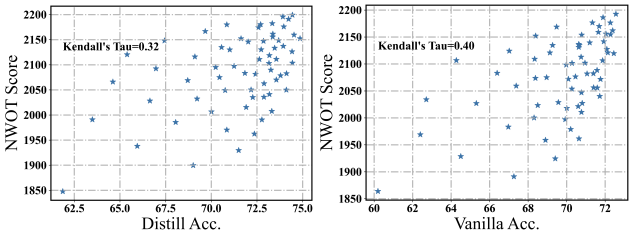

Figure 1. Left: Correlation of distill accuracy & NWOT score on search space S_0. Right: Correlation of vanilla accuracy & NWOT score on search space S_0.

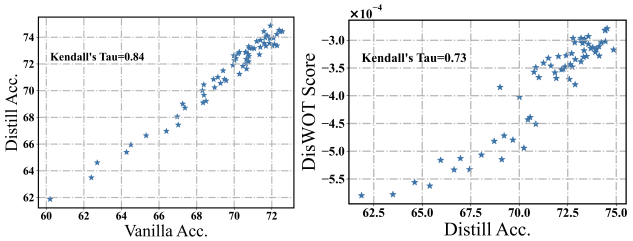

Figure 2. Left: Correlation of vanilla accuracy & distill accuracy on search space S_0. Right: Correlation of distilling accuracy & DisWOT score on search space S_0.

large discrepancy between them, with an 8% difference in Kendall's Tau. Figure 2 (left) demonstrates the ranking correlation between vanilla accuracy and distillation accuracy. Surprisingly, their correlation is only 0.84, which indicates that there is a non-negligible gap between distillation results and vanilla results. On the one hand, it indicates that a new zero-cost metric needs to be designed to improve the ranking consistency for the distillation. On the other hand, vanilla accuracy can be used as a rough measure of ranking consistency when distillation accuracy is unavailable. Figure 2 (right) illustrates that our proposed DisWOT achieve better ranking consistency of distillation accuracy on search space S_0.

Analysis of detailed examples. Table 3 summarizes 4 groups of specific student pairs with vanilla-distill gaps. For groups A and B, despite student models A2, B2 having more parameters and better vanilla accuracy, their distillation accuracy is inferior to A1, B1. This indicates an important effect on the overall depth of students for distillation. For groups C and D, the results show that the model with more blocks in stage-2 enjoys better distillation accuracy. In addition, DisWOT predicts the correct scores for these models.

Table 3. Parameters (K), vanilla accuracy (%), distillation accuracy (%), and prediction scores (10^{-4}) of DisWOT for the student on search space S_0.

Group	Student	Param.	Vanilla Acc.	Distill Acc.	DisWOT
A1	ResNet[7,1,3]	259.89	69.13	71.01	4.41
A2	ResNet[3,3,3]	278.32	69.57	70.76	3.34
B1	ResNet[7,5,3]	334.13	70.76	72.58	5.44
B2	ResNet[1,7,3]	343.22	70.77	72.18	5.20
C1	ResNet[5,5,7]	620.72	71.93	74.86	7.37
C2	ResNet[3,7,7]	648.50	72.45	74.42	7.33
D1	ResNet[7,3,5]	444.98	72.04	73.36	4.85
D2	ResNet[5,5,5]	472.76	72.09	73.94	8.17

A.4. Semantic properties of random networks

DisWOT leverages the semantic similarity metric of a randomly initialized teacher-student model to predict distillation performance, abandoning the training-based NAS paradigm. Models with various architectures have different semantic features because of their different effective receptive fields. To represent semantic and localization information, Gradient-weighted Class Activation Mapping (Grad-CAM [25]) methods have been widely adopted in weakly supervised object localization and model interpretability [2]. Recently, several studies [2, 28] reveal that randomly initialized models also have favorable semantic localization capabilities. As shown in the visual localization heatmaps of Figure 3, we can intuitively observe that randomly initialized networks can locate a single object without any training. The visualization results demonstrate that semantic information exists even in random networks, which can localize the objects in an image. In addition, we evaluate different strategies (e.g. CAM, Grad-CAM, and SCDA [28]) for semantic localization maps and find that Grad-CAM achieves stable prediction performance for the semantic similarity metric. Thus, we adopt the Grad-

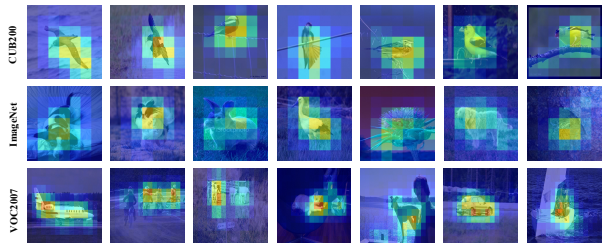

Figure 3. Localization results of a randomly initialized network on ImageNet, VOC2007, and CUB-200. The network can localize the objects in an image, with a small standard deviation between different trials. Note that this figure is from Tobias [3].

CAM of deeper layers to capture the informative relation similarity in this paper.

A.5. Comparisons with different NAS approaches.

We presents the results of more NAS methods in this section and find that our proposed DisWOT is better able to distinguish good architectures than both One-shot NAS and Zero-shot NAS. As shown in Table 7 and 4, we compare DisWOT with one-shot NAS (e.g. ENAS [23], SETN [6], SPOS [7]) and zero-shot NAS(a.k.a. Zen-NAS [19]). Results demonstrate that DisWOT achieves better performance than its counterparts in distillation and classification results. In addition, we conduct more correlation evaluation in Tab. 5. The results show that DisWOT surpasses other proxies in the NAS-Bench-101/101-KD/201-KD and DisWOT ($\mathcal{M}_s$) achieves superior correlations than DisWOT ($\mathcal{M}_r$), which are consistent with findings in space S_0.

Table 4. Top-1 accuracy (%) of different NAS algorithms under distill training on NAS-Bench-201 [5].

Datasets	ENAS [23]	SETN [6]	SPOS [7]	Zen-NAS [19]	DisWOT
CIFAR-10	34.94	81.61	92.98	89.60	**93.55**
CIFAR-100	11.14	59.78	72.91	71.86	**74.21**
ImageNet16-120	11.53	29.91	43.50	39.44	**47.30**

A.6. About KD-based zero-cost proxies.

In this section, we present DisWOT as a new universal zero proxy and propose a series of KD-based zero proxies based on this motivation. As shown in Table 6, we further provide the ranking correlation of various KD-based zero-cost proxies on three datasets. We adopt the optimal architecture in the search space of NAS-Bench-201 as a teacher network and conduct 10 independent experiments of three knowledge distillation methods (a.k.a., CC [22], KD [10], and NST [13]). We observe that DisWOT achieves an acceptable ranking correlation on three datasets. NST [13] show impressive ranking ability, which is the best KD-based zero-proxies in DisWOT framework. DisWOT reveals better performance than most of zero-cost proxies under vanilla training, as shown in Table 7.

Table 5. "mean±std %" Spearman correlation on NAS-Bench-101 and NAS-Bench-201. NAS-Bench-101/201-KD denotes to distill accuracies of the architectures on NAS-Bench-101/201.

Method	NAS-Bench-101	NAS-Bench-201	NAS101-KD	NAS201-KD
FLOPs	30.81%±0.00	63.38%±0.06	15.56%±0.04	64.55%±0.01
Fisher	-38.81%±0.14	35.91%±0.09	-33.92%±0.14	4.45%±0.08
Grad_Norm	-39.23%±0.08	58.70%±0.11	-39.16%±0.01	-10.01%±0.11
SNIP	-29.01%±0.09	58.17%±0.15	-21.78%±0.02	16.91%±0.10
Synflow	43.69%±0.12	**74.61%±0.08**	20.36%±0.08	74.63%±0.02
NWOT	32.84%±0.51	64.41%±0.08	22.97%±0.04	35.27%±0.03
DisWOT ($\mathcal{M}_s$)	**49.61%±0.05**	65.74%±0.07	50.16%±0.09	53.88%±0.06
DisWOT ($\mathcal{M}_r$)	30.74%±0.06	56.46%±0.08	42.94%±0.11	45.27%±0.07

Table 6. Ranking correlation of our KD-based zero-cost proxies on NAS-Bench-201.

Datasets	Method	Kendall's Tau	Spearman	Pearson
CIFAR-10	CC [22]	0.48	0.68	0.56
	KD [10]	0.35	0.50	0.40
	NST [13]	0.64	0.83	0.72
	DisWOT	0.41	0.61	0.54
CIFAR-100	CC [22]	0.43	0.65	0.58
	KD [10]	0.38	0.54	0.55
	NST [13]	0.57	0.72	0.64
	DisWOT	0.56	0.72	0.65
ImageNet16	CC [22]	0.53	0.71	0.66
	KD [10]	0.44	0.61	0.65
	NST [13]	0.54	0.74	0.74
	DisWOT	0.49	0.69	0.55

Table 7. Top-1 accuracy (%) of different NAS algorithms under vanilla training on NAS-Bench-201 [5].

Datasets	ENAS [23]	SETN [6]	SPOS [7]	Zen-NAS [19]	DisWOT
CIFAR-10	53.89	87.64	93.23	90.70	**93.37**
CIFAR-100	13.96	59.05	71.03	68.26	**71.53**
ImageNet16-120	14.84	32.52	42.19	40.60	**45.50**

B. Details of Search Space and Settings

In this section, we introduce the implementation details in the three search spaces and the detailed training settings.

B.1. S_0 search space

Search space. As illustrated in Figure 8, we construct the search space S_0 based on ResNet20, a simple resnet designed for CIFAR-10/100, where each building block consists of two 3×3 convolutional layers and the depth of each residual block is searched in set {1,3,5,7}. The search space size is $4^3 = 64$ in total.

Implementation details. As for results of vanilla classification results, we train each architecture in the search space S_0 with the same strategy. For each architecture in the search space S_0, we adopt ResNet110 as a teacher network. Specifically, we train each architecture via momentum SGD, using

Table 8. Supernet architecture of the S_0 search space. Each line describes a sequence of 1 or more identical layers, repeated *repeat* times. All layers in the same sequence have the same number of output channels.

input	block	channels	repeat	stride
$32^2 \times 3$	3×3 conv	16	1	2
$32^2 \times 16$	Res Block	16	[1,3,5,7]	2
$16^2 \times 16$	Res Block	32	[1,3,5,7]	2
$8^2 \times 32$	Res Block	64	[1,3,5,7]	2
$8^2 \times 64$	Global Avgpool	-	1	-
64	FC	100	1	-

cross-entropy loss for 240 epochs. We set the weight decay as 5e-4 and adopted a multi-stage scheduler to decay the learning rate from 0.1 to 0. We use the random flip with the probability of 0.5, the random crop 32×32 patch with 4 pixels paddings, and the normalization over RGB channels. All of the experiments are based on CIFAR-100 datasets. As for the distillation results, the vanilla knowledge distillation methods [10, 15–18, 30, 30] are adopted. Specifically, we conduct experiments based on the CRD [26]. For KD [10], we follow the Equation 1 and set $\alpha = 0.9$ and $\rho = 4$.

$$\mathcal{L}_{KL} = \alpha\rho^2 CE\left(\sigma\left(z^T/\rho\right), \sigma\left(z^S/\rho\right)\right) \tag{1}$$

where z^T and z^S denote the logits of teacher and student, respectively. ρ is the temperature, α is a balancing weight, and σ is a softmax function. CE denotes the cross entropy loss.

B.2. S_1 search space

Search space. The search space S_1 is following the cell-based search space NAS-Bench-201 [5], where a cell is represented as a directed acyclic graph (DAG). Each edge in search space S_1 is associated with an operation selected from a predefined operation set, which consists of (1) zero, (2) skip connection, (3) 1×1 convolution, (4) 3×3 convolution, and (5) 3×3 average pooling layer. The DAG has 4 nodes, each representing the sum of all features from previous nodes. The search space size of S_1 is 15,625 in total.

Implementation Details. We randomly sampled 50 candidate architectures to evaluate ranking consistency. All experiments are implemented on a single NVIDIA 3090Ti GPU, with the baseline from the AutoDL [5]. We recommend using a network with higher complexity or better performance in the search space as the teacher network. The process of DisWOT is divided into three steps: (1) Determine a specific teacher network (deeper or more complex). (2) Perform an evolutionary search with DisWOT metrics to obtain the best student network. (3) Distill the student network with vanilla KD [11] based on a specific teacher network. The distillation setting is the same as Section B.1.

Searched architectures. For other NAS methods, the searched architectures (see Table 9) of RS [5], ENAS [23], SETN [6] and SPOS [4, 7, 12, 29] are borrowed from the official implementation [5], and the remaining zero-shot NAS methods utilize the same evolutionary search algorithm. Expressly, we set the initial population size as 20, and the sample size as 10. The total evolution search cycle is set as 5,000. We calculate the zero-proxy score with only one batch of data as fitness during evolution. The teacher network used in DisWOT is the best architecture in the search space.

B.3. S_2 search space

Search space. Following NDS [24], we design the search space for CIFAR and ImageNet, respectively. The search space designed for CIFAR consists of a stem, followed by 6 stages, and a head, as shown in Table 10. The i-th stage consists of d_i blocks with c_i channels and stride of $s_i \in \{1, 2\}$. The number of channels c_i needs to be divisible by 8, and the minimal number of channels should be larger than 8. The candidate blocks can be residual blocks or bottleneck blocks defined in ResNet, and the kernel size can be chosen from set {3,5,7}. As shown in Table 11, the search space designed for ImageNet consists of 4 stages, following the configuration of ResNet18.

Table 10. Design space parameterization of S_2 for CIFAR-10/100. "POOL" denotes the global average pooling, and "FC" denotes a fully connected network.

stage	block	channels	repeat	stride
steam	3×3 conv	c_0	1	1
stage1	{block}	c_1	d_1	s_1
stage2	{block}	c_2	d_2	s_2
stage3	{block}	c_3	d_3	s_3
stage4	{block}	c_4	d_4	s_4
stage5	{block}	c_5	d_5	s_5
stage6	{block}	c_6	d_6	s_6
head	POOL + FC	10	-	

Table 11. Design space parameterization of S_2 for ImageNet. "POOL" denotes the global average pooling, and "FC" denotes fully connected network.

stage	block	channels	repeat	stride
steam	7×7 conv	c_0	1	2
stage1	{block}	c_1	d_1	s_1
stage2	{block}	c_2	d_1	s_2
stage3	{block}	c_3	d_1	s_3
stage4	{block}	c_4	d_1	s_4
head	POOL + FC	1000	-	-

Training settings. We search the ResNet18 level network

Table 9. Searched architectures of NAS algorithms. The searched results are denoted by a string from NAS-Bench-201 API [5].

	Searched architectures
RS [5]	|skip_connect~0|+|nor_conv_3x3~0|skip_connect~1|+|nor_conv_3x3~0|nor_conv_1x1~1|avg_pool_3x3~2|
ENAS [23]	|skip_connect~0|+|avg_pool_3x3~0|skip_connect~1|+|avg_pool_3x3~0|skip_connect~1|skip_connect~2|
SETN [6]	|nor_conv_3x3~0|+|skip_connect~0|skip_connect~1|+|skip_connect~0|skip_connect~1|avg_pool_3x3~2|
SPOS [7]	|skip_connect~0|+|nor_conv_1x1~0|nor_conv_3x3~1|+|nor_conv_1x1~0|avg_pool_3x3~1|nor_conv_3x3~2|
Zen-NAS [19]	|skip_connect~0|+|nor_conv_3x3~0|nor_conv_3x3~1|+|skip_connect~0|skip_connect~1|nor_conv_3x3~2|
NWOT [21]	|nor_conv_1x1~0|+|nor_conv_3x3~0|nor_conv_1x1~1|+|nor_conv_1x1~0|nor_conv_3x3~1|nor_conv_1x1~2|
DisWOT(ours)	|skip_connect~0|+|nor_conv_3x3~0|nor_conv_1x1~1|+|nor_conv_1x1~0|nor_conv_3x3~1|nor_conv_3x3~2|

regarding the search space in NDS [24]. Specifically, we limit the number of parameters to less than 13M and the depth of the network to up to 20 layers and find the optimal network by evolution algorithm with the DisWOT metric. Please refer to Section C.1 for more details about the evolutionary search for the search space S_2. Specifically, we adopt ResNet34 as a teacher network and conduct a vanilla knowledge distillation process [11] with $\rho = 1$ and $\alpha = 3$ as shown in Equation 1. For ImageNet, we follow the standard PyTorch practice, and the batch size is 256.

Searched architectures. After the evolutionary search, we presented the optimal student network obtained for CIFAR and ImageNet, as shown in the Table 12 and Table 13, respectively. We observe that the searched architecture of DisWOT has very different characteristics from the artificially designed student architecture, i.e., DisWOT prefers student networks with larger convolutional kernels in shallow layers. Generally speaking, teacher networks tend to have deeper layers and thus have a larger receptive field. Guided by the teacher network, DisWOT favors larger convolutional kernels in the shallow layers so that the receptive field of the student network is as close to that of the teacher as possible. However, the network searched on ImageNet only changed the number of channels under the parameter restriction. We infer that expanding the kernel size leads to a massive amount of additional parameters, which will lead to exceeding the budget. We infer that the network will prefer a larger kernel if a sufficient budget is available.

Table 12. Search results of the ResNet-like search space for CIFAR-10/100. "Basic" denotes the basic block proposed in ResNet [9].

input	block	channels	repeat	stride
$32^2 \times 3$	3×3 conv	88	1	1
$32^2 \times 88$	Basic 7×7	96	3	1
$32^2 \times 120$	Basic 5×5	192	2	2
$16^2 \times 192$	Basic 5×5	176	2	1
$16^2 \times 96$	Basic 5×5	168	3	2
$8^2 \times 168$	Basic 3×3	112	3	2
$4^2 \times 112$	Basic 3×3	512	1	1
512	POOL + FC	1000	1	-

Table 13. Search results of the ResNet-like search space for ImageNet under 13M parameter limit. "Basic" denotes the basic block proposed in ResNet [9].

input	block	channels	repeat	stride
$224^2 \times 3$	7×7 conv	96	1	2
$112^2 \times 96$	Basic 3×3	64	3	2
$56^2 \times 64$	Basic 3×3	128	2	2
$28^2 \times 128$	Basic 3×3	256	2	2
$14^2 \times 256$	Basic 3×3	512	2	2
512	POOL + FC	1000	1	-

Table 14. DisWOT-11.7M based on ImageNet (Searched for segmentation task.)

block	kernel	in	out	stride	bottleneck	# layers
Conv	7	3	64	2	-	1
Res	3	64	64	2	64	2
Res	3	64	128	2	128	2
Res	3	128	256	2	256	2
Res	3	256	512	2	512	2
Conv	1	512	2384	1	-	1

C. Details of Algorithm for DisWOT

In this section, we describe the implementation details of the evolutionary alorhigtm and the implementation code of DisWOT.

C.1. Implementation of Evolutionary Algorithm

Here we adopt Evolutionary Algorithm (EA) as an architecture generator to find optimal student network. In this section, we further describe the mutation process of the evolutionary algorithm in details. In the evolutionary algorithm, we randomly generate P architectures with constraints C and then select the $top - k$ architectures by DisWOT metric from the population. Then we randomly select the parent architecture from the $top - k$ architectures and mutate it.

The mutation algorithm is presented in Algorithm 1. Specifically, for S_2 search space, we provide basic blocks with a kernel size of {3,5,7} and choose {0.67,0.8,1.25,1.5} as the mutation range for channels. The number of channels should be divisible by 8, and the max number of channels is 2048. The depth of chosen block can be mutated in range {+1, -1}. After mutating the architecture, we check whether it is valid, e.g., the parameter is meed the predefined constraint.

Algorithm 1 Mutation Algorithm for DisWOT

Input: Parent architecture A_i, Search space S.
Output: Mutated architecture $\hat{P}_i$

1: Randomly select a block a_i from Parent architecture A_i;
2: Randomly mutate the kernel size of a_i from $S(\{block\})$;
3: Randomly mutate the width of a_i from $S(c_i)$;
4: Randomly mutate the depth of a_i from $S(d_i)$;
5: Check whether the mutated architecture is valid;
6: Return the mutated architecture $\hat{P}_i$;

C.2. Implementation of metric in DisWOT

The section presents the implementation of semantic similarity metric and relation similarity metric in DisWOT. The semantic similarity metric measures the inter-correlation on the accumulated Grad-CAM for teacher and student networks, whose calculation needs one forward and one backward to get the localization information. The relation similarity metric measures the relationship between input samples whose activations of teacher and student networks are needed.

Implementation of semantic similarity metric. Here, we present the implementation code of the relation similarity metric, as shown in List C.2. We need the Grad-CAM maps of all classes for calculation, which needs at least one forward and one backward to get the Grad-CAM similarity. Different from ICKD [20], there are mainly two differences: (1) The Gaussian initialized teacher network and student network is backpropagated only once. (2) We only use the grad of the fully-connected layer for calculation.

Listing 1. The PyTorch implementation of semantic similarity metric.

```
import torch
import torch.nn as nn
import torch.nn.functional as F

def semantic_similarity_metric(teacher,
    student, batch_data):
   criterion = nn.CrossEntropyLoss()
   image, label = batch_data
   # Forward once.
   t_logits = teacher.forward(image)
   s_logits = student.forward(image)
   # Backward once.
   criterion(t_logits, label).backward()
   criterion(s_logits, label).backward()
   # Grad-cam of fc layer.
   t_grad_cam = teacher.fc.weight.grad
   s_grad_cam = student.fc.weight.grad
   # Compute channel-wise similarity
   return -1 *
       channel_similarity(t_grad_cam,
       s_grad_cam)

def channel_similarity(f_t, f_s):
   bsz, ch = f_s.shape[0], f_s.shape[1]
   # Reshape
   f_s = f_s.view(bsz, ch, -1)
   f_t = f_t.view(bsz, ch, -1)
   # Get channel-wise similarity matrix
   emd_s = torch.bmm(f_s, f_s.permute(0,
       2, 1))
   emd_s = F.normalize(emd_s, dim=2)
   emd_t = torch.bmm(f_t, f_t.permute(0,
       2, 1))
   emd_t = F.normalize(emd_t, dim=2)
   # Produce L_2 distance
   G_diff = emd_s - emd_t
   return (G_diff * G_diff).view(bsz,
       -1).sum() / (ch * bsz)
```

Implementation of relation similarity metric. Here we present the implementation code of relation similarity, as shown in List C.2. With only the logits of the teacher and student network, our relation similarity metric is easy to implement. There are mainly two differences compared with SP [27]: (1) The teacher and student networks are initialized with kaiming initialization [8], which means the teacher network did not undergo any backpropagation. (2) Here, we only used the activation before global average pooling as input, and the activations of shallow layers are not utilized. In fact, we find that activations closer to the output are more informative. When using activations from shallow layers, experiments demonstrate that the relation similarity ranks poorly.

Listing 2. The PyTorch implementation of relation similarity metric.

```
import torch
import torch.nn as nn
import torch.nn.functional as F

def relation_similarity_metric(teacher,
    student, batch_data):
   image, label = batch_data
   # Forward pass
   t_feats =
       teacher.forward_features(image)
   s_feats =
       student.forward_features(image)
```

```
    # Get activation before average pooling
    t_feat = t_feats[-2]
    s_feat = s_feats[-2]
    # Compute batch similarity
    return -1 * batch_similarity(t_feat,
        s_feat)

def batch_similarity(f_t, f_s):
    # Reshape
    f_s = f_s.view(f_s.shape[0], -1)
    f_t = f_t.view(f_t.shape[0], -1)
    # Get batch-wise similarity matrix
    G_s = torch.mm(f_s, torch.t(f_s))
    G_s = F.normalize(G_s)
    G_t = torch.mm(f_t, torch.t(f_t))
    G_t = F.normalize(G_t)
    # Produce L_2 distance
    G_diff = G_t - G_s
    return (G_diff * G_diff).view(-1,
        1).sum() / (bsz * bsz)
```

References

[1] Mohamed S Abdelfattah, Abhinav Mehrotra, Łukasz Dudziak, and Nicholas Donald Lane. Zero-cost proxies for lightweight nas. In ICLR, 2020. 2

[2] Wonho Bae, Junhyug Noh, and Gunhee Kim. Rethinking class activation mapping for weakly supervised object localization. In ECCV, 2020. 2

[3] Yun-Hao Cao and Jianxin Wu. A random cnn sees objects: One inductive bias of cnn and its applications. In AAAI, 2022. 3

[4] Peijie Dong, Xin Niu, Lujun Li, Zhiliang Tian, Xiaodong Wang, Zimian Wei, Hengyue Pan, and Dongsheng Li. Rd-nas: Enhancing one-shot supernet ranking ability via ranking distillation from zero-cost proxies. arXiv preprint arXiv:2301.09850, 2023. 4

[5] Xuanyi Dong and Yi Yang. Nas-bench-201: Extending the scope of reproducible neural architecture search. In ICLR, 2019. 3, 4, 5

[6] Xuanyi Dong and Yezhou Yang. One-shot neural architecture search via self-evaluated template network. 2019 ICCV, 2019. 3, 4, 5

[7] 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. arXiv preprint arXiv:1904.00420, 2019. 3, 4, 5

[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. 6

[9] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In CVPR, 2016. 1, 5

[10] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. In arXiv:1503.02531, 2015. 1, 3, 4

[11] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. arXiv preprint arXiv:1503.02531, 2015. 4, 5

[12] Yiming Hu, Xingang Wang, Lujun Li, and Qingyi Gu. Improving one-shot nas with shrinking-and-expanding supernet. Pattern Recognition, 2021. 4

[13] Zehao Huang and Naiyan Wang. Like what you like: Knowledge distill via neuron selectivity transfer. arXiv:1707.01219, 2017. 3

[14] Namhoon Lee, Thalaiyasingam Ajanthan, and Philip Torr. Snip: Single-shot network pruning based on connection sensitivity. In ICLR, 2018. 2

[15] Lujun Li. Self-regulated feature learning via teacher-free feature distillation. In ECCV, 2022. 4

[16] Lujun Li and Zhe Jin. Shadow knowledge distillation: Bridging offline and online knowledge transfer. In NeuIPS, 2022. 4

[17] Lujun Li, Liang Shiuan-Ni, Ya Yang, and Zhe Jin. Boosting online feature transfer via separable feature fusion. In IJCNN, 2022. 4

[18] Lujun Li, Yikai Wang, Anbang Yao, Yi Qian, Xiao Zhou, and Ke He. Explicit connection distillation. In ICLR, 2020. 4

[19] Ming Lin, Pichao Wang, Zhenhong Sun, Hesen Chen, Xiuyu Sun, Qi Qian, Hao Li, and Rong Jin. Zen-nas: A zero-shot nas for high-performance image recognition. 2021. 3, 5

[20] Li Liu, Qinwen Huang, Sihao Lin, Hongwei Xie, Bing Wang, Xiaojun Chang, and Xiao-Xue Liang. Exploring inter-channel correlation for diversity-preserved knowledge distillation. 2021 ICCV, 2021. 6

[21] Joe Mellor, Jack Turner, Amos Storkey, and Elliot J Crowley. Neural architecture search without training. In ICML, 2021. 1, 2, 5

[22] Baoyun Peng, Xiao Jin, Jiaheng Liu, Dongsheng Li, Yichao Wu, Yu Liu, Shunfeng Zhou, and Zhaoning Zhang. Correlation congruence for knowledge distillation. In ICCV, 2019. 3

[23] Hieu Pham, Melody Y. Guan, Barret Zoph, Quoc V. Le, and Jeff Dean. Efficient neural architecture search via parameter sharing. In ICML, 2018. 3, 4, 5

[24] Ilija Radosavovic, Justin Johnson, Saining Xie, Wan-Yen Lo, and Piotr Dollár. On network design spaces for visual recognition. 2019 ICCV, 2019. 4, 5

[25] R. R. Selvaraju, Abhishek Das, Ramakrishna Vedantam, Michael Cogswell, D. Parikh, and Dhruv Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. 2019. 2

[26] Yonglong Tian, Dilip Krishnan, and Phillip Isola. Contrastive representation distillation. In ICLR, 2020. 4

[27] Frederick Tung and Greg Mori. Similarity-preserving knowledge distillation. In ICCV, 2019. 6

[28] Xiu-Shen Wei, Jian-Hao Luo, Jianxin Wu, and Zhi-Hua Zhou. Selective convolutional descriptor aggregation for fine-grained image retrieval. TIP, 2017. 2

[29] Zimian Wei, Hengyue Pan, Lujun Li Li, Menglong Lu, Xin Niu, Peijie Dong, and Dongsheng Li. Convformer: Closing the gap between cnn and vision transformers. arXiv preprint arXiv:2209.07738, 2022. 4

[30] Liu Xiaolong, Li Lujun, Li Chao, and Anbang Yao. Norm: Knowledge distillation via n-to-one representation matching. In ICLR, 2023. 4