SasWOT: Real-Time Semantic Segmentation Architecture Search WithOut Training

Chendi Zhu[1*], **Lujun Li**[2*], **Yuli Wu**[3], **Zhengxing Sun** [1†]

[1]State Key Laboratory for Novel Software Technology, Nanjing University
[2]The Hong Kong University of Science and Technology
[3]Institute of Imaging and Computer Vision, RWTH Aachen University, Aachen, Germany
chendi.zhu@smail.nju.edu.cn, lilujunai@gmail.com, yuli.wu@lfb.rwth-aachen.de, szx@nju.edu.cn

Abstract

In this paper, we present SasWOT, the first training-free Semantic segmentation Architecture Search (SAS) framework via an auto-discovery proxy. Semantic segmentation is widely used in many real-time applications. For fast inference and memory efficiency, Previous SAS seeks the optimal segmenter by differentiable or RL Search. However, the significant computational costs of these training-based SAS limit their practical usage. To improve the search efficiency, we explore the training-free route but empirically observe that the existing zero-cost proxies designed on the classification task are sub-optimal on the segmentation benchmark. To address this challenge, we develop a customized proxy search framework for SAS tasks to augment its predictive capabilities. Specifically, we design the proxy search space based on the some observations: (1) different inputs of segmenter statistics can be well combined; (2) some basic operators can effectively improve the correlation. Thus, we build computational graphs with multiple statistics as inputs and different advanced basis arithmetic as the primary operations to represent candidate proxies. Then, we employ an evolutionary algorithm to crossover and mutate the superior candidates in the population based on correlation evaluation. Finally, based on the searched proxy, we perform the segmenter search without candidate training. In this way, SasWOT not only enables automated proxy optimization for SAS tasks but also achieves significant search acceleration before the retrain stage. Extensive experiments on Cityscapes and CamVid datasets demonstrate that SasWOT achieves superior trade-off between accuracy and speed over several state-of-the-art techniques. More remarkably, on Cityscapes dataset, SasWOT achieves the performance of 71.3% mIoU with the speed of 162 FPS.

Introduction

Semantic segmentation predicts pixel-level annotations of different semantic categories for an image. Recently, many real-time applications like autonomous driving require segmenter to fast inference on low-power edge devices and guarantee satisfied performance. However, many state-of-the-art models stack convolutions, fuse multi-scale features and increase the resolution to improve accuracy, making them suffer from high computational resources and memory budget (Liu et al. 2023; Li et al. 2023b; Li and Jin 2022; Li et al. 2023a, 2022b,a, 2020; Li 2022; Shao et al. 2023). To tackle this challenge, there are many lightweight architectural engineering(Badrinarayanan, Kendall, and Cipolla 2017; Paszke et al. 2016) proposed to allow the segmenter to deploy on resource-constrained platforms. For example, ICNet (Zhao et al. 2018a) uses an image cascade network to incorporate multi-resolution inputs. BiSeNet (Yu et al. 2018a), and DFANet allocate more computing resources to feature fusion modules and utilize lightweight backbones. However, this architectural engineering also requires extensive expert design and lots of experimental trial and error.

To address this issue, some semantic segmentation architecture search (SAS) methods have been presented by building search spaces and processes. For example, Auto-DeepLab (Liu et al. 2019) first searches cell structures and the downsampling strategy to optimize resolutions. CAS (Zhang et al. 2019) searches for operators and decoders in

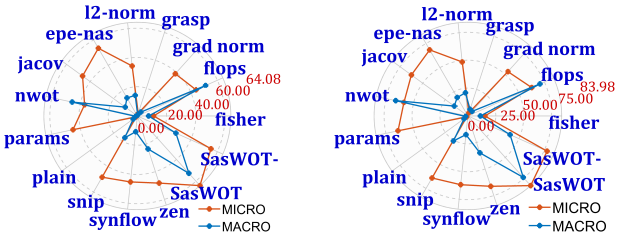

Figure 1: Kendall-Tau (left) and Spearman (right) Correlation on Segmentation Benchmark.

Method	Algorithm	Cost
Auto-DeepLab (Liu et al. 2019)	Gradient	72 h
GAS (Li et al. 2019a)	Graph	160 h
CAS (Zhang et al. 2019)	Gradient	200 h
DF-Seg (Li et al. 2019b)	Pruning	9600 h
FasterSeg (Chen et al. 2020)	Gradient	48 h
SasWOT	Training-free	1.8 h

Table 1: Comparison of various SAS methods. h indicates GPU-hour.

*These authors contributed equally.
†Corresponding author.

a pre-defined multi-scale network with customized latency constraints. Although these methods achieve promising results, they also introduce significant computational overhead and optimization difficulties during the search phase. As shown in Table 1, these methods always involve training a supernet from scratch that is larger than the original model to perform a differentiable or Reinforcement Learning (RL) search. Thus, they need lots of costs for search, which brings heavy burdens for practical usage.

As Albert Einstein once said: *"Everything should be made as simple as possible, but not simpler"*. Inspired by the recent zero-shot NAS, we experimented with training-free methods for classification tasks and evaluated their predictability on a segmentation benchmark. As shown in Figure 1, these methods typically perform worse on segmentation tasks. These failures are large because they are hand-designed fixed forms that do not generalize well to new segmentation tasks with different data domains, architectural styles, and evaluation metrics. To design customized proxies for segmentation tasks, we analyze the existing methods and discover that: (1) Proxies with different model statistic inputs can be well combined. (Tanaka et al. 2020) This indicates that we can design proxies to combine various inputs to improve predictability. (2) Most proxies can be factored into some input options and basic operations. For example, log, abs, and matrix multiplication are often present in proxy expressions (Tanaka et al. 2020; Lee, Ajanthan, and Torr 2018; Lin et al. 2021). This indicates that the proxy can also be designed from scratch following Auto-Zero.

Based on the above observations, we present, SasWOT, an automated search framework that utilizes evolutionary algorithms to search training-free proxies for SAS efficiently. Firstly, we represent the proxy search space with the segmenter's multiple statistics (*e.g.*, weights and gradient) as input and different unary and binary mathematical operations as candidates. Next, our search algorithm initializes the population, evaluates, crosses, and varies for a better proxy. During the search, we directly use the ranking correlation with accuracy results in the segmentation benchmark as optimization objectives to find a more fitting proxy for the segmentation task. To speed up the process, we employ judgment and elitism-preserve strategies during the proxy search. With the automatic search framework, our SasWOT surpasses existing training-free NAS approaches by a large margin without prior knowledge. Finally, we search for segmentation architectures with SasWOT and then implement a complete training process on the searched segmentation architecture.

We perform extensive experiments to validate the performance and efficiency strengths of our SasWOT in semantic segmentation benchmarks and on multiple autonomous driving datasets. On the segmentation benchmark, SasWOT consistently improves ranking consistency compared to other state-of-the-art training-free methods, on Cityscapes and CamVid benchmarks, our method yields search acceleration of at least 30 times compared to the traditional gradient-based SAS method and allows the segmenter search to be completed on a single GPU within 2 hours.

The main contributions can be summarized as follows:

- We propose a novel real-time segmenter search framework via auto-discovery proxy without training, which to our knowledge, is the first training-free search method for segmentation.
- We present a comprehensive proxy search space and evolve proxy with correlation in segmentation as fitting objectives. In addition, we achieve significant search acceleration by searching for segmenters with discovered proxies.
- We conduct extensive experiments on the standard Cityscapes and CamVid benchmarks. Compared to other real-time methods, our method obtains a state-of-the-art trade-off between performance and latency.

Related Work

Semantic Segmentation Architecture Search Following FCN (Long, Shelhamer, and Darrell 2015), many advanced Handcrafted architectures are presented for larger receptive field (Zhao et al. 2017; Chen et al. 2017a,b, 2018; Yu et al. 2018b) and better pixel-wise relationships (Zhao et al. 2018b; Huang et al. 2019; Fu et al. 2019; Song et al. 2019) to improve the performance of semantic segmentation. To alleviate resource budget issues, some light-weighted Segmenter like ENet (Paszke et al. 2016), ICNet (Zhao et al. 2018a) and BiSeNet (Yu et al. 2018a) have been proposed in real-time applications. However, manual design methods are hard to achieve a good trade-off between performance and efficiency, so researchers develop semantic segmentation architecture search to automatically optimize the segmenter. For example, Auto-DeepLab (Liu et al. 2019) pioneered in searching cell-level and network-level dense-connected search space to achieve better spatial resolution changes. CAS (Zhang et al. 2019) first imposed resource constraints while searching for an efficient backbone for segmentation. Subsequent SAS methods present advanced search space designs (*e.g.*, multi-resolution branch) and search algorithms (*e.g.*, gradient, pruning, meta-learning). However, these training-based methods always require training the supernetwork to evaluate the performance of different candidates, resulting in complex optimization processes and large additional computational overheads. To address this issue, we propose SasWOT, the first training-free framework for SAS. Our SasWOT directly uses proxy scoring on well-initialized segmenters to search without any training cost. Our SasWOT not only opens new doors for SAS research, but also dramatically improves the search efficiency in practical applications.

Training-Free Architecture Search Traditional architecture search involves designing search spaces, search algorithms, and evaluation strategies to automatically discover the optimal architecture within certain constraints (Wei et al. 2024; Hu et al. 2021; Dong et al. 2022; Yang et al. 2022; Dong, Li, and Wei 2023; Dong et al. 2023; Lu et al. 2024; Zimian Wei et al. 2024). The training-based methods employ a train-then-search process by multiple trials or weight-sharing policies. Training-free architecture search, also known as zero-shot NAS, is a faster alternative to vanilla NAS and one-shot NAS, as it can predict network

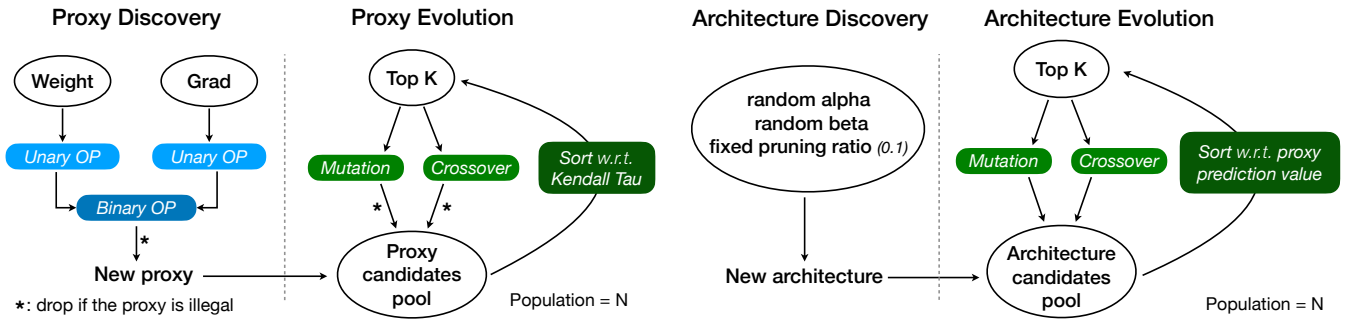

Figure 2: The overall process of SasWOT includes automated proxy discovery and training-free architecture search. In the proxy search phase, we build candidates with gradients and weights as inputs and different unary binary operations as options. we perform an evolutionary search to remove weak individuals and crossover & mutation to generate new populations within promising ones. Finally, we pick up the best-performing proxy for training-free architecture search, which evolves different architectures in the search space using scores of SasWOT proxy.

performance without training network parameters. Zero-cost proxies can be divided into parameter-level and architecture-level proxies. Parameter-level zero-cost proxies rely on pruning and summing up the saliency value of each layer weight as the proxy. Several methods have been proposed for Convolution-based architectures, such as Synflow (Tanaka et al. 2020), SNIP (Lee, Ajanthan, and Torr 2018), and GraSP (Wang, Zhang, and Grosse 2020), which rely on gradient computations using a single minibatch of data at initialization and can be computed very quickly. Architecture-level zero-cost proxies evaluate the network's discriminability from the architecture level. Zen-NAS (Lin et al. 2021) proposed a novel zero-shot proxy called Zen-Score to evaluate the expressivity of a network, while NWOT (Mellor et al. 2021) examined how the linear maps induced by data points correlate for untrained network architectures. However, these methods all employ hand-designed fixed proxies for the classification task and encoder-only CNN models. In semantic segmentation, the encoder-decoder model usually has complex branches and multi-scale features that result in huge challenges for traditional proxies in predicting final accuracy. To address this challenge, we automatically optimize the first customized proxies for SAS based on some observations. Our SasWOT first explore the training-free SAS method and bridges theoretical proxy design and practical downstream applications.

Methodology

Our framework is divided into two parts: (1) evolving customized proxies on a semantic segmentation benchmark (2) evolving optimal segmenters utilizing searched proxies. In this section, we first illustrate the proxy search component on search space design, search process, and search results. Then, we introduce the details of our trainning-free segmenter search. The pipeline of our approach is shown in Figure 2).

Search Space for Proxy Discovery

Search Space Organization Our approach evolves the customized proxy in a semantic segmentation benchmark (Duan et al. 2021), which consists of U-Net-like encoder-decoder models with various operators and their individual training results. We extract the activations (A), gradients(G), and weights(W) of each convolutional layer in both encoder and decoder modules as input. Then, we evaluate the existing proxy formulation with these inputs, and observe that: (1) Although with the same input statistics, different mathematical operations can bring large performance disparities for proxies. This point suggests that we can improve the proxy by tuning it with different mathematical operators. (2) Proxy models with different inputs can be well combined. For example, Synflow and NWOT can be well integrated with better correlation. This motivates us to select multiple statistic inputs for the proxy search space, bringing additional gains. Based on the above observations, we include activations, gradients, and weights as multiple inputs to the search and represent the zero-cost proxy as a computation graph. Then we collect the common mathematical operators for intermediate nodes of the graph, which are detailed in the next part.

Primitive Operations In the context of the zero-cost proxy, primitive operations are used to process segmenter statistics, resulting in a computational graph for performance evaluation. We consider two types of primitive operations, including unary operations (operations with only one operand) and binary operations (operations with two operands). A summary of available primitive operations is presented as follows:

- Unary operations: "elementwise pow,""to-sum-scalar," "elementwise exp,""normalize,""elementwise relu," "elementwise sign,""elementwise invert,""slogdet," "frobenius norm,""elementwise normalized sum," "l1norm,""softmax,""sigmoid,""logsoftmax," "to-mean-scalar,""gram-matrix,""elementwise log," "elementwise abslog,""elementwise abs,""no op".
- Binary operations: "elementwise sum,""equal to," "elementwise difference,""elementwise product," "lesser than,""greater than,""matrix multiplication," "hamming distance,""pairwise distance,""l1 loss,"

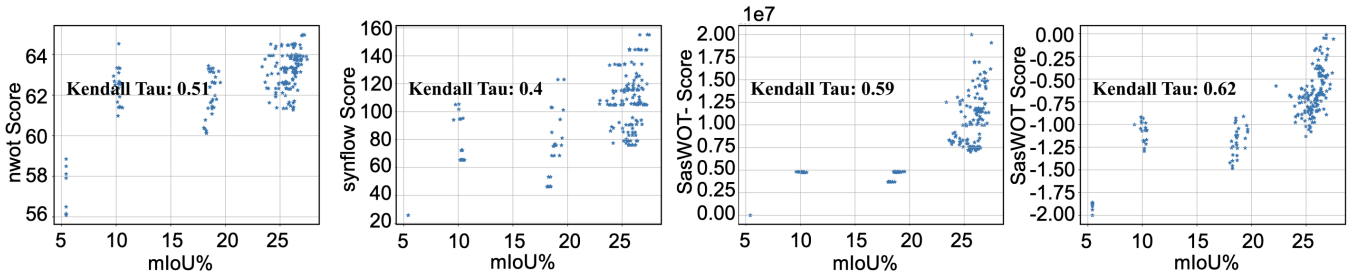

Figure 3: Correlation visualization of NWOT, Synflow, SasWOT-, SasWOT (from left to right).

"cosine similarity,""kl divergence,""mse loss".

These mathematical operations are essential building blocks to construct a diverse and expressive search space of zero-cost proxy. We provide more details of their formulas and properties in the supplementary material.

Evolution Procedure, Acceleration, and Objective

Based on our proxy search space and graph representation, we leverage an evolutionary algorithm (EA) to search for optimal proxy expressions. Our EA starts with an initial population of candidate Proxies and iteratively evolves the population over generations using genetic operators, such as selection, crossover, and mutation, to generate better solutions. For fitting objectives, each proxy is evaluated based on the ranking correlation between its output proxy $\mathcal{Q}$ scores and ground truths $\mathcal{D}$ to efficiently discover the optimal proxy ρ^* from search space $\mathcal{P}$, as:

$$\rho^*_{SasWOT} = \arg\max_{\rho \in \mathcal{P}} (\tau(\mathcal{D}, \mathcal{Q})). \quad (1)$$

where Kendall's Tau τ is formulated as the correlation coefficient. Each evolution picks top-k candidates with the highest evaluation score and randomly selects a parent from the candidates for mutation. To improve the efficiency of gene evolution, we adopt a greedy strategy to maximize the evolution of candidates. Genetic algorithms are designed to iteratively evolve the best-fit candidates.Greed is demonstrated by picking only topK candidates from the candidates pool after mutation and crossover operators, instead of randomly picking from the candidates pool in each round as in traditional genetic algorithms. Only the topK of evolved candidates is retained in each round, and if the number of candidates is less than the population at that point, new proxies are randomly sampled, and the topK is drawn, and the candidate pool is emptied before the start of the next round. In the mutation step, depending on the mutation mode, we randomly select unary or binary operations to change. In the crossover step, we randomly select primitive operations from the two proxies to form a new proxy. To improve the search efficiency of the proxy, we propose **Early-Rejection Strategy**. If two unary operators cannot perform a binary operation due to a dimensional mismatch, the genetic algorithm will immediately eliminate the combination. Also, if no better candidates are produced after five iterations, the topK at that point will be recorded and the candidates pool will be reset randomly. Once a candidate proxy has optimization collapse and overflow issues, this strategy will directly break its verification and then generates new offspring to avoid searching for local optimums.

Algorithm 1: Evolution Search for SasWOT Proxy

Input: Search space $\mathcal{S}$, population $\mathcal{P}$, max iteration $\mathcal{T}$, sample ratio r, sampled pool $\mathcal{R}$, topk k, .
Output: SasWOT proxy with best KD(Kendall Tau) index.

```
P0 := Initialize population(P_i);
Sample pool R := ∅;
for i = 1, 2, ..., T do
    Select G_i^s := GetTopk(R, k);
    // greedy evolution strategy
    Clear sample pool R := ∅;
    Mutate := MUTATE(G_i^s);
    Crossover G_i^c := CROSSOVER(G_i^s);
    Append G_i^m to R;
    Append G_i^c to R;
    if R < P then
        Add random samples
    else
        Go to line 4;
    end if
end for
```

Searched Zero-cost Proxy

In table 3, by considering the correlation of zero-cost proxies in MICRO and MACRO space, we decided to try to combine two relatively well-performing proxies (NWOT, SasWOT-) after normalization. The experimental results (SasWOT) show that this was a successful attempt. The comparison between SasWOT- and SasWOT highlights that AVT-normalized proxy combination strategies can achieve better results than individually searched proxies. We present formulas of the searched proxies SasWOT- and SasWOT as follows:

$$\rho^*_{SasWOT-} = \sum_{i=1}^{N_{cl}} (softmax(W_{cl_i}) < ReLU\left(\frac{\partial \mathcal{L}}{\partial W_{cl_i}}\right)) \quad (2)$$

$$\rho^*_{SasWOT} = norma_{avt}(\rho^*_{SasWOT-}) + norma_{avt}(log|K_H|) \quad (3)$$

where N_{cl} is the number of convolution and linear layers, W_{cl_i} is the weight parameter matrix of a convolution layer or a linear layer, $\partial\mathcal{L}/\partial W_{cl_i}$ is the corresponding gradient matrix. K_H is the kernel matrix of binary codes in

Method	InputSize	mIoU (%)		FPS	FLOPs	Params	Search Method	Training cost	Search cost
		val	test					(GPU days)	(GPU hours)
Enet	640×360	-	58.3	76.9	3.8G	0.4M	Manual	-	-
BiSeNet	768×1536	69	68.4	105.8	14.8G	5.8M	Manual	-	-
Fast-SCNN	1024×2048	68.6	68	123.5	-	1.1M	Manual	-	-
ICNet	1024×2048	-	69.5	37.7	28.3G	26.5M	Manual	-	-
DFANet	1024×1024	-	71.3	100	3.4G	7.8M	Manual	-	-
SFNet(DF1)	1024×2048	-	74.5	NA	-	9.03M	Manual	-	-
GAS	769×1537	-	71.8	108.4	-	-	Graph	-	160
CAS	768×1536	71.6	70.5	108	-	-	Gradient	-	200
DF1-Seg-d8	1024×2048	72.4	71.4	136.9	-	-	Pruning	-	9600
FasterSeg	1024×2048	69.8	-	163.9	28.03G	3.42M	Gradient	3.24	48
Random	1024×2048	67.9	65.8	162.37	29.17G	4.64M	Training-free	2.63	1
NWOT	1024×2048	69.2	68.4	162.61	30.56G	2.29M	Training-free	2.54	1
Synflow	1024×2048	67.0	66.0	161.13	33.42G	3.88M	Training-free	2.75	1
SasWOT-	1024×2048	69.2	66.7	162.04	32.13G	3.12M	Training-free	2.58	1
SasWOT	1024×2048	71.3	69.8	162.64	29.34G	3.33M	Training-free	2.55	1.8

Table 2: mIoU and inference FPS on Ciytscapes validation (val) and test (test) sets. The training and search costs are obtained from the original papers (Enet(Paszke et al. 2016), BiSeNet (Yu et al. 2018a), Fast-SCNN (Poudel, Liwicki, and Cipolla 2019), ICNet (Zhao et al. 2018a), DFANet A (Zhang et al. 2023), SFNet(DF1) (Li et al. 2022c), GAS (Li et al. 2019a) , CAS (Zhang et al. 2019) , DF1-Seg-d8 (Li et al. 2019b) and FasterSeg (Chen et al. 2020)) report or from our experimental records.

NWOT (Mellor et al. 2021).

SasWOT is built on the combination of two proxies, one from the SasWOT- obtained from the search and the other from the proxy proposed by the previous work. Considering the magnitude of different proxies, we adopt the strategy of AVT normalization to achieve the summation between two proxies. Given a set of N_{arch} architectures L, let one proxy predict the result as $P_{zc}(L)^{1\times N_{arch}}$, then the AVT normalization of proxy prediction $p_{zc}(L_i)$ for one architecture is given by:

$$norm_{avt}(p_{zc}(L_i)) = \frac{p_{zc}(L_i) - \min(P_{zc}(L))}{\max(P_{zc}(L)) - \min(P_{zc}(L))} \quad (4)$$

In our experiments, we find that the combination of SasWOT- and NWOT (Mellor et al. 2021) has an effective improvement in evaluation metrics. However, due to the static nature of the AVT normalization policy, we are unable to score the architecture in real-time in practice. To address this issue, we estimate the maximum and minimum values of possible SasWOT scores by pre-selecting a certain number of architectures in the search space at random. Figure 3 has demonstrated the superiority of SasWOT- and SasWOT, which significantly outperform the previous zero-cost proxy methods and meanwhile enjoy more search efficiency.

Training-free Segmenter Search

Our approach utilizes a training-free segmenter search to efficiently explore a large search space of candidate architectures without the need for costly and time-consuming training on large datasets. To begin, we employ an evolutionary search algorithm to obtain a good proxy model. Once we have a reliable proxy, we conduct a training-free segmenter search by using the evolutionary algorithm to discover the optimal segmenter α^* from the search space $\mathcal{A}$. While evolutionary search is typically the most effective approach, we opt for an EA search in this case due to the limited size of the search space, which simplifies the process. Using random initialized weights W, we conduct the training-free search algorithm to identify the optimal Segmenter efficiently, as:

$$\alpha^* = \arg\max_{\alpha\in\mathcal{A}} \rho^*_{SasWOT}(\alpha, \mathcal{W}). \quad (5)$$

In the architecture search process, we first randomly generate matrices that can denote the model architectures and add them to the candidate pool. Then the architectures are scored by zero-cost predictor and the top K architectures are selected for the mutation and crossover steps. In the mutation step, depending on the mutation pattern, we randomly change the rows of the alpha or beta matrix. In the crossover step, we randomly select the alpha and beta matrices from the two architectures to generate a new architecture. After the mutation and crossover steps, if the population is insufficient, new alpha and beta matrices are randomly generated and added to the candidates. the new architectures in each of these steps need to be verified for legitimacy, i.e., whether they have been recorded before.

Experiments

In this section, we first evaluate the ranking performance of the searched proxies in the semantic segmentation benchmark, and then we use the searched proxies to perform a training-free segmenter search in Cityscapes (Cordts et al. 2016) and CamVid (Brostow et al. 2008). All models are trained from scratch without ImageNet pre-trained weights. In all experiments, the class mIoU (mean Intersection over Union per class) and FPS (frame per second) are used as the metrics for accuracy and speed, respectively.

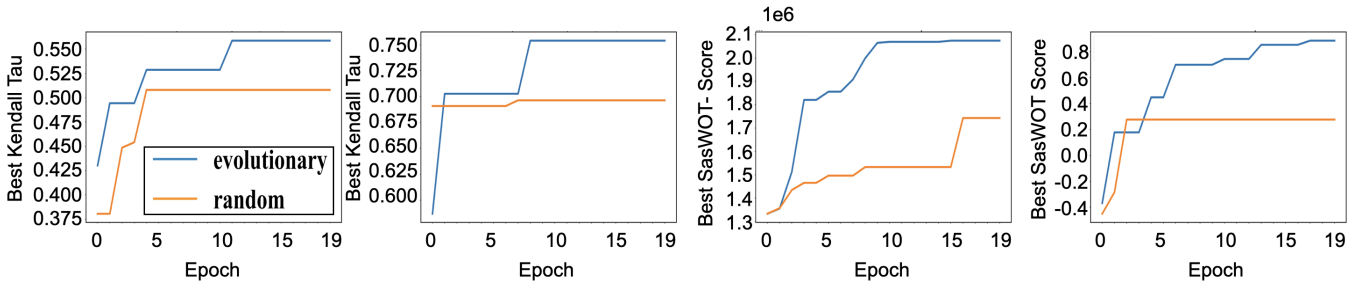

Figure 4: Visualization of search curves. From left to right: proxy search in macro search space, proxy search in micro search space, segmenter search with SasWOT- proxy, segmenter search with SasWOT proxy.

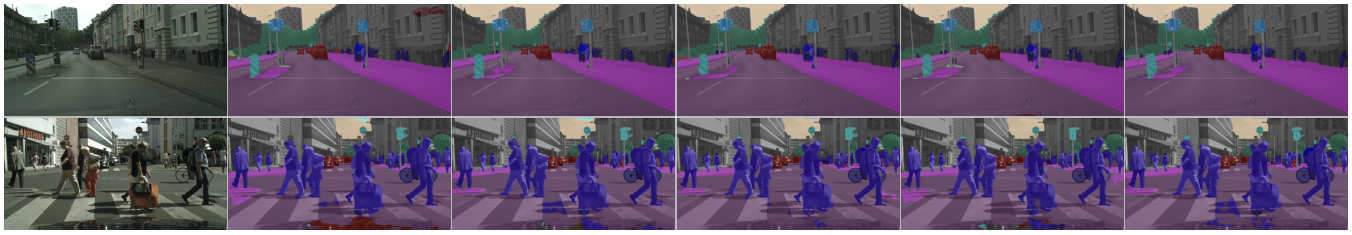

Figure 5: Visualization of segmentation results on CityScapes. From left to right:ground truth, (architecture searched with) random, nwot, synflow, SasWOT-, SasWOT

Space	MICRO			MACRO		
Ranking	Kd	Sp	Ps	Kd	Sp	Ps
	Mean	Mean	Mean	Mean	Mean	Mean
Fisher	11.11	15.20	10.84	7.66	11.17	5.99
FLoPs	43.98	62.55	53.12	51.18	70.69	46.72
Grad_norm	38.83	53.49	22.48	3.62	5.49	4.42
Grasp	1.90	3.02	3.02	0.66	6.28	4.18
Epe_nas	53.48	68.40	66.29	14.26	19.15	10.71
Jacov	46.45	62.53	43.64	10.31	15.08	19.80
NWOT	36.7	58.52	52.09	45.71	66.09	58.40
Params	45.11	63.96	58.27	1.22	2.05	1.56
SNIP	48.34	64.17	35.47	17.17	25.88	24.38
Synflow	45.36	61.94	51.83	10.85	16.82	18.40
Zen-NAS	48.27	66.27	43.64	23.73	34.78	35.05
SasWOT	64.08	83.98	76.62	52.76	74.06	67.97
SasWOT-	55.22	77.63	75.51	28.92	41.34	37.06

Table 3: Ranking results (%) on TransNAS-Bench-101 (Duan et al. 2021)-Semantic Segmentation. Kd: Kendall Tau, Sp: spearman, Ps: Pearson.

Method	mIoU (%)	FPS	FLOPs
Random	62.2	388.8	7.28G
NWOT(Mellor et al. 2021)	62.2	392.5	7.63G
Synflow(Tanaka et al. 2020)	57.1	390.3	8.34G
SasWOT-	58.7	388.9	8.03G
SasWOT	64.3	388.7	7.33G

Table 4: Results on the CamVid test set with resolution 960 ×720.

Experiments on Segmentation Benchmark

Dataset and implementation. The TransNAS-Bench-101 (Duan et al. 2021) dataset was used as our segmentation benchmark to evaluate the performance of the searched proxies. This dataset includes 3256 and 4096 networks from the macro-level search space and the cell-level search space. These networks were trained for 30 epochs with the same setup on the Taskonomy dataset. The Taskonomy dataset was sampled from 17 classes of MSCOCO(Lin et al. 2014). The Labels for the Taskonomy dataset were predicted from a network pre-trained on the MSCOCO dataset. We tested the performance of the searched proxies in micro and macro search spaces, respectively. Fifty architectures were randomly selected for evaluation in each experiment, with the Kendall Tau index, Spearman index, and Pearson index being evaluated. Through several sets of experiments, we found that the proxy SasWOT- found by evolution search had a more stable model performance prediction, as its average Kendal Tau index was higher compared to the other proxies. SasWOT further improves the performance of SasWOT-, which is more evident in the Micro search space.

Comparison results Table 3 demonstrates the ranking capabilities of different methods on macro and micro search spaces. These results demonstrate that (1) our SasWOT achieves the best ranking performance and outperforms other proxy methods, (2) SasWOT- performs well in some cases, indicating that our single-input statistic proxy search is effective, and SasWOT improves on this by demonstrating the advanced nature of our multi-input strategy, (3) for other methods, NWOT, Synflow, and Flops have some advantages over other methods. This may be attributed to the properties of the search space.

Experiments on Cityscapes

Dataset and implementation. We evaluate SasWOT on Cityscapes dataset(Cordts et al. 2016). The dataset includes over 5,000 images, each with high-quality pixel-level annotations for various semantic segmentation tasks. Inherited from Fasterseg's architecture search space, SasWOT enables two branches to share three operators. Unlike previous work, SasWOT fixes the pruning rate of the searched student model, thus ensuring a reduced model complexity. In the retraining phase, we adopted a knowledge distillation strategy and used a deeplabv3+ model with a backbone of resnet101 as the teacher network. For this task, we used an SGD optimizer with an initial learning rate of 0.015 and an exponential learning rate decay. We trained the searched architectures for 800 epochs and evaluated SasWOT on the Cityscapes validation and test sets. We used a raw image resolution of 1024×2048 (H×W) to measure mIoU and velocity inference.

Comparison results In Table 2, we see the performance metrics and inference speed of the models searched using different proxies. Compared to the proxies presented in previous work, the architecture search using saswot- achieves no less than the architecture searched by NWOT. It is worth mentioning that SasWOT further improves the performance of the searched architectures on top of SasWOT- with no additional computational cost increase. It achieves a performance comparable to the gradient-based Fasterseg search method but with a significant reduction (about 24x faster) in search time. In addition, we also compared other training-free methods utilizing the same search and training settings. The experimental results show that NWOT is a superior method among them. In conclusion, SasWOT can achieve a significant improvement in search efficiency compared to the previous SAS methods and superior performance compared to other training-free methods on Cityscapes.

Experiments on CamVid

Dataset and implementation. CamVid is another street scene dataset extracted from five video sequences taken from a driving automobile. It contains 701 images in total, where 367 for training, 101 for validation and 233 for testing. The images have a resolution of 720×960 (H×W) and 11 semantic categories. Considering the spatial resolution of the Fasterseg search space, we cropped the input images randomly and then resized them to 512×1024 (H×W). Similar to experiments on cityscapes, we used a deeplabv3+ teacher network for knowledge distillation with the searched architectures. We trained the searched architecture 80 epochs using the SGD optimizer, with an initial learning rate of 0.01 and exponential learning rate decay.

Comparison results Table 4 reveals that by achieving this metric in only 80 training epochs and maintaining an inference speed that approximates that of Fasterseg, the SasWOT search framework is certainly more efficient. The teacher network deeplabv3+ only achieves comparable segmentation results to SasWOT with the same training setup while consuming several times the computational power. The implication of this finding is that the SasWOT search framework may be a more practical and efficient approach to semantic segmentation tasks, especially in real-time applications where inference speed is critical. However, it is important to note that the comparison is specific to the CamVid dataset and may not necessarily generalize to other datasets or tasks.

Ablation Study

Search algorithm. The evolutionary search algorithm used in this study is a gradient-free optimization technique that is commonly used in AutoML. As shown in Figure 4, the evolutionary algorithm obtains faster convergence and better final search results compared to random search in the proxy and architecture search process. This suggests that the evolutionary algorithm is a more effective and efficient approach to our SasWOT than random search.

Correlation Visualization. Figure 3 reports that the score of SasWOT and the performance ground-truth are visualized to observe the proxy's predictability. The performance ground-truth represents the actual segmentation results obtained by the larger and more complex segmentation model. The figure shows that SasWOT can effectively detect the actual segmentation results, indicating that it is a reliable proxy model for semantic segmentation tasks.

Segmentation Result Visualization. As shown in Figure 5, the visualization reveals that our method can accurately segment some detailed regions that are challenging for other methods, such as small objects and thin structures. Additionally, our method can effectively segment some important categories, e.g., vehicles and pedestrians, which are critical for applications like autonomous driving and surveillance. The visualization also demonstrates that our method can maintain region connectivity, meaning that adjacent regions of the same semantic class are accurately grouped together.

Conclusion

In this paper, we present a novel training-free SAS framework, dubbed SasWOT, to efficiently search for the optimal segmenter utilizing our automated search proxy. Our framework includes customized proxy search and training-free segmenter search. With the discovered proxies, our sasWOT allows an efficient search for promising candidates without any training cost. As a result, our SasWOT achieves at least $30 \times$ acceleration in the search stage. Comprehensive experiment results on segmentation benchmarks and multiple autonomous driving segmentation datasets illustrate the superior ranking ability improvement and segmentation performance of our method. We hope that our novel investigations would give more insight and new directions for the semantic segmentation and NAS research community.

Acknowledgments

The paper is supported by: The National Natural Science Foundation of China No. 42075139,42077232; The Science and technology program of Jiangsu Province No. BE2020082 and BE2022063; The Innovation Fund of State Key Laboratory for Novel Software Technology No. ZZKT2022A18.

References

Badrinarayanan, V.; Kendall, A.; and Cipolla, R. 2017. Segnet: A deep convolutional encoder-decoder architecture for image segmentation. *IEEE transactions on pattern analysis and machine intelligence*, 39(12): 2481–2495.

Brostow, G. J.; Shotton, J.; Fauqueur, J.; and Cipolla, R. 2008. Segmentation and recognition using structure from motion point clouds. In *European conference on computer vision*, 44–57. Springer.

Chen, L.-C.; Papandreou, G.; Kokkinos, I.; Murphy, K.; and Yuille, A. L. 2017a. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *T-PAMI*.

Chen, L.-C.; Papandreou, G.; Schroff, F.; and Adam, H. 2017b. Rethinking atrous convolution for semantic image segmentation. *arXiv preprint arXiv:1706.05587*.

Chen, L.-C.; Zhu, Y.; Papandreou, G.; Schroff, F.; and Adam, H. 2018. Encoder-decoder with atrous separable convolution for semantic image segmentation. In *Proceedings of the European conference on computer vision (ECCV)*, 801–818.

Chen, W.; Gong, X.; Liu, X.; Zhang, Q.; Li, Y.; and Wang, Z. 2020. FasterSeg: Searching for Faster Real-time Semantic Segmentation. In *International Conference on Learning Representations*.

Cordts, M.; Omran, M.; Ramos, S.; Rehfeld, T.; Enzweiler, M.; Benenson, R.; Franke, U.; Roth, S.; and Schiele, B. 2016. The cityscapes dataset for semantic urban scene understanding. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 3213–3223.

Dong, P.; Li, L.; and Wei, Z. 2023. DisWOT: Student Architecture Search for Distillation WithOut Training. In *CVPR*.

Dong, P.; Li, L.; Wei, Z.; Niu, X.; Tian, Z.; and Pan, H. 2023. EMQ: Evolving Training-free Proxies for Automated Mixed Precision Quantization. *arXiv preprint arXiv:2307.10554*.

Dong, P.; Niu, X.; Li, L.; Xie, L.; Zou, W.; Ye, T.; Wei, Z.; and Pan, H. 2022. Prior-Guided One-shot Neural Architecture Search. *arXiv preprint arXiv:2206.13329*.

Duan, Y.; Chen, X.; Xu, H.; Chen, Z.; Liang, X.; Zhang, T.; and Li, Z. 2021. Transnas-bench-101: Improving transferability and generalizability of cross-task neural architecture search. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 5251–5260.

Fu, J.; Liu, J.; Tian, H.; Li, Y.; Bao, Y.; Fang, Z.; and Lu, H. 2019. Dual attention network for scene segmentation. In *CVPR*.

Hu, Y.; Wang, X.; Li, L.; and Gu, Q. 2021. Improving one-shot NAS with shrinking-and-expanding supernet. *Pattern Recognition*.

Huang, Z.; Wang, X.; Huang, L.; Huang, C.; Wei, Y.; and Liu, W. 2019. Ccnet: Criss-cross attention for semantic segmentation. In *ICCV*.

Lee, N.; Ajanthan, T.; and Torr, P. H. 2018. Snip: Single-shot network pruning based on connection sensitivity. *arXiv preprint arXiv:1810.02340*.

Li, G.; Qian, G.; Delgadillo, I. C.; Müller, M.; Thabet, A.; and Ghanem, B. 2019a. SGAS: Sequential Greedy Architecture Search. *ArXiv*.

Li, L. 2022. Self-Regulated Feature Learning via Teacher-free Feature Distillation. In *ECCV*.

Li, L.; Dong, P.; Li, A.; Wei, Z.; and Ya, Y. 2023a. KD-Zero: Evolving Knowledge Distiller for Any Teacher-Student Pairs. In *Thirty-seventh Conference on Neural Information Processing Systems*.

Li, L.; Dong, P.; Wei, Z.; and Yang, Y. 2023b. Automated Knowledge Distillation via Monte Carlo Tree Search. In *ICCV*.

Li, L.; and Jin, Z. 2022. Shadow knowledge distillation: Bridging offline and online knowledge transfer. *Advances in Neural Information Processing Systems*.

Li, L.; Shiuan-Ni, L.; Yang, Y.; and Jin, Z. 2022a. Boosting Online Feature Transfer via Separable Feature Fusion. In *IJCNN*.

Li, L.; Shiuan-Ni, L.; Yang, Y.; and Jin, Z. 2022b. Teacher-free Distillation via Regularizing Intermediate Representation. In *IJCNN*.

Li, L.; Wang, Y.; Yao, A.; Qian, Y.; Zhou, X.; and He, K. 2020. Explicit Connection Distillation.

Li, X.; Zhang, J.; Yang, Y.; Cheng, G.; Yang, K.; Tong, Y.; and Tao, D. 2022c. Sfnet: Faster, accurate, and domain agnostic semantic segmentation via semantic flow. *arXiv preprint arXiv:2207.04415*.

Li, X.; Zhou, Y.; Pan, Z.; and Feng, J. 2019b. Partial order pruning: for best speed/accuracy trade-off in neural architecture search. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 9145–9153.

Lin, M.; Wang, P.; Sun, Z.; Chen, H.; Sun, X.; Qian, Q.; Li, H.; and Jin, R. 2021. Zen-NAS: A Zero-Shot NAS for High-Performance Image Recognition.

Lin, T.-Y.; Maire, M.; Belongie, S.; Hays, J.; Perona, P.; Ramanan, D.; Dollár, P.; and Zitnick, C. L. 2014. Microsoft coco: Common objects in context. In *European conference on computer vision*, 740–755. Springer.

Liu, C.; Chen, L.-C.; Schroff, F.; Adam, H.; Hua, W.; Yuille, A. L.; and Fei-Fei, L. 2019. Auto-deeplab: Hierarchical neural architecture search for semantic image segmentation. In *CVPR*.

Liu, X.; Li, L.; Li, C.; and Yao, A. 2023. NORM: Knowledge Distillation via N-to-One Representation Matching. *arXiv preprint arXiv:2305.13803*.

Long, J.; Shelhamer, E.; and Darrell, T. 2015. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 3431–3440.

Lu, L.; CHen, Z.; Lu, L., Xiaoyu; Rao, Y.; Li, L.; and Pang, S. 2024. UniADS: Universal Architecture-Distiller Search for Distillation Gap. In *AAAI*.

Mellor, J.; Turner, J.; Storkey, A.; and Crowley, E. J. 2021. Neural architecture search without training. In *ICML*.

Paszke, A.; Chaurasia, A.; Kim, S.; and Culurciello, E. 2016. Enet: A deep neural network architecture for real-time semantic segmentation. *arXiv preprint arXiv:1606.02147*.

Poudel, R. P.; Liwicki, S.; and Cipolla, R. 2019. Fast-SCNN: fast semantic segmentation network. *arXiv preprint arXiv:1902.04502*.

Shao, S.; Dai, X.; Yin, S.; Li, L.; Chen, H.; and Hu, Y. 2023. Catch-Up Distillation: You Only Need to Train Once for Accelerating Sampling. *arXiv preprint arXiv:2305.10769*.

Song, L.; Li, Y.; Li, Z.; Yu, G.; Sun, H.; Sun, J.; and Zheng, N. 2019. Learnable Tree Filter for Structure-preserving Feature Transform. In *NeurIPS*.

Tanaka, H.; Kunin, D.; Yamins, D. L.; and Ganguli, S. 2020. Pruning neural networks without any data by iteratively conserving synaptic flow. *NeurIPS*.

Wang, C.; Zhang, G.; and Grosse, R. 2020. Picking winning tickets before training by preserving gradient flow. *arXiv preprint arXiv:2002.07376*.

Wei, Z.; Pan, H.; Li, L. L.; Lu, M.; Niu, X.; Dong, P.; and Li, D. 2024. TVT: Training-free Vision Transformer Search on Tiny Datasets. In *ICASSP)*.

Yang, C.; Zhou, H.; An, Z.; Jiang, X.; Xu, Y.; and Zhang, Q. 2022. Cross-Image Relational Knowledge Distillation for Semantic Segmentation. *arXiv preprint arXiv:2204.06986*.

Yu, C.; Wang, J.; Peng, C.; Gao, C.; Yu, G.; and Sang, N. 2018a. Bisenet: Bilateral segmentation network for real-time semantic segmentation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 325–341.

Yu, C.; Wang, J.; Peng, C.; Gao, C.; Yu, G.; and Sang, N. 2018b. Learning a discriminative feature network for semantic segmentation. In *CVPR*.

Zhang, Y.; Li, K.; Zhang, G.; Zhu, Z.; and Wang, P. 2023. DFA-UNet: Efficient Railroad Image Segmentation. *Applied Sciences*, 13(1): 662.

Zhang, Y.; Qiu, Z.; Liu, J.; Yao, T.; Liu, D.; and Mei, T. 2019. Customizable Architecture Search for Semantic Segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 11641–11650.

Zhao, H.; Qi, X.; Shen, X.; Shi, J.; and Jia, J. 2018a. Icnet for real-time semantic segmentation on high-resolution images. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 405–420.

Zhao, H.; Shi, J.; Qi, X.; Wang, X.; and Jia, J. 2017. Pyramid scene parsing network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2881–2890.

Zhao, H.; Zhang, Y.; Liu, S.; Shi, J.; Change Loy, C.; Lin, D.; and Jia, J. 2018b. Psanet: Point-wise spatial attention network for scene parsing. In *ECCV*.

Zimian Wei, Z.; Li, L. L.; Dong, P.; Hui, Z.; Li, A.; Lu, M.; Pan, H.; and Li, D. 2024. Auto-Prox: Training-Free Vision Transformer Architecture Search via Automatic Proxy Discovery. In *AAAI*.