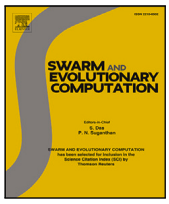
Swarm and Evolutionary Computation 93 (2025) 101837

Contents lists available at ScienceDirect

Swarm and Evolutionary Computation

journal homepage: www.elsevier.com/locate/swevo



Mde-EvoNAS: Automatic network architecture design for monocular depth estimation via evolutionary neural architecture search

Zhihao Yu [a], Haoyu Zhang [a,*], Ruyu Liu [a], Sheng Dai [b], Xinan Chen [c], Weiguo Sheng [a], Yaochu Jin [d]

[a] School of Information Science and Technology, Hangzhou Normal University, Hangzhou, 310000, China
[b] Department of Colorectal Surgery, Sir Run Run Shaw Hospital, School of Medicine, Zhejiang University, Hangzhou, Zhejiang, 310016, China
[c] Digital Port Technologies Lab, School of Computer Science, University of Nottingham Ningbo China, Ningbo, 315100, China
[d] Trustworthy and General AI Lab, School of Engineering, Westlake University, Hangzhou, 310030, China

ARTICLE INFO

Keywords:
Evolutionary algorithm
Neural architecture search
Monocular depth estimation
Multiscale channel-spatial feature awareness module
SuperNet
Intestinal tissue depth estimation

ABSTRACT

The advanced performance of the monocular depth estimation model highly relies on features extracted by encoder networks. The encoder architecture in most previous methods reuses networks designed for image classification. It might be sub-optimal because of the gap between the tasks of monocular depth estimation and image classification. However, manually designing task-specific encoders is difficult and tedious for users who lack extensive experience in deep learning. To address this problem, we propose a computationally efficient evolutionary neural architecture search framework, which can automatically design encoders tailored to specific monocular depth estimation tasks. To improve the search efficiency of evolutionary optimization, we construct the search space as a supernet based on the technique of one-shot NAS. In each generation, the supernet is stochastically trained based on the parent population, and each offspring individual inherits weights from the supernet for direct fitness evaluation. Subsequently, we introduce the multiscale spatial feature awareness module within the monocular depth estimation model to leverage channel-wise relationships and positional information derived from multiscale feature maps, enhancing the feature representations of objects across various scales. The experiment results demonstrate the superior performance of our method in both depth benchmark datasets (i.e., KITTI, NYU-Depth v2) and intestinal tissue depth estimation.

1. Introduction

Many real-world applications, such as autonomous driving, robotic navigation, and augmented reality, heavily depend on depth information, where precise depth perception plays a critical role in improving safety, efficiency, and decision-making [1,2]. Early depth estimation techniques often relied on binocular vision [3], where two cameras captured images from slightly different viewpoints to triangulate the depth of objects in a scene. These methods can provide accurate depth information but come with several limitations, such as the need for precise camera calibration, the complexity of handling occlusions, and the requirement for two cameras, which increases the cost of the system [4,5]. In recent years, there has been a growing trend towards monocular depth estimation, which uses a single camera to infer depth. Compared with binocular vision systems, the monocular systems are more cost-effective and easier to deploy [6]. Hence, research on depth estimation from a single monocular image using the deep learning technique has attracted widespread attention in the industry and academia [7].

Monocular depth estimation is the process of predicting depth information from a single image. It uses machine learning techniques to infer how far objects are from the camera based on visual cues in the image. Hence, in addition to the higher-level contextual information that is solely sufficient for image classification, monocular depth estimation requires feature maps that capture texture details, geometric structures, and lighting variations from the image. Standard deep learning-based Monocular depth estimation typically follows the encoder–decoder convention shown in Fig. 1. A Convolutional Neural Network (CNN) encoder is used to first extract features, followed by another decoder to aggregate these features and output a prediction of the depth of each pixel. The performance of monocular depth estimation models highly relies on features extracted by encoders. However, many monocular depth estimation models directly use networks designed for image classification as encoders. It might be sub-optimal because the difference between these tasks calls for different network architecture design principles. Such distinctions make network architectures designed for image classification tasks fall short. Therefore, customizing

* Corresponding author.
E-mail address: Haoyu.Zhang@hznu.edu.cn (H. Zhang).

https://doi.org/10.1016/j.swevo.2024.101837
Received 10 September 2024; Received in revised form 20 December 2024; Accepted 25 December 2024
Available online 7 January 2025

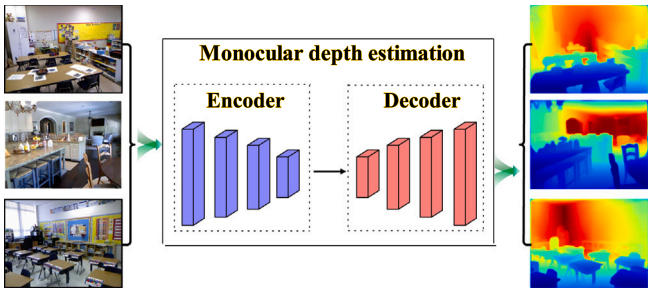


Fig. 1. Overview of standard monocular depth estimation model.

the encoder architecture for different scenarios is crucial for improving the performance of monocular depth estimation models.

However, designing a task-specific neural network with promising performance, extensive expertise in both the investigated problem domain and deep learning is required, which is not necessarily available to every interested user. Recently, the NAS technique has presented a promising approach to address this problematic procedure by considering the network architecture design for the target task as an optimization problem [8]. With the recent algorithmic development of NAS on image classification [9–11], objective detection [12,13], and semantic segmentation [14], searched networks have surpassed the performance of the manually designed network. While these recent results have established a promising starting point for computer vision tasks, the potential of NAS has yet to be fully explored in more challenging tasks, such as monocular depth estimation.

Mainstream NAS algorithms typically rely on reinforcement learning (RL), gradient-based (GD), or evolutionary computation (EC) to design neural network architectures. RL-based NAS automates the process of designing neural network structures by using a learning agent to iteratively explore and optimize architectures for improved performance [15,16]. However, RL-based NAS methods need to optimize the learning agent policy iteratively, which results in high computational complexity [17]. Another large body of research on NAS uses GD methods to search for a good network [18–20]. In the above work, the search space is reformulated via continuous relaxation, which transfers the discrete search space to a continuous space. Then, the architecture parameters can be optimized using the GD-based method. Although the search efficiency has been dramatically improved from the continuous relaxation trick, the GD-based method tends to bias networks with faster convergence, leading to potentially sub-optimal networks being searched by the algorithm, as faster converging networks need not necessarily be the ones that also generalize better [21].

Concurrently, EC-based NAS (EvoNAS) algorithms have earned a plethora of attention in the deep learning community [22–25]. ECs have the advantage of exploring a broad solution space through mechanisms like mutation and crossover, allowing them to find diverse solutions. They are also insensitive to local minimum and less reliant on gradient information, making them suitable for problems where gradient-based methods struggle. Despite the population-based nature of EC enabling effectiveness in addressing various complex optimization problems, direct porting ideas from the current algorithm of ECs would not suffice the requirements of NAS for monocular depth estimation.

The main challenges of applying the EvoNAS approach for monocular depth estimation are two aspects: (1) The computational budget caused by the fitness evaluations of candidate networks in every generation can be prohibitively expensive [26]. As a result, it is impractical to apply the EvoNAS method to large-scale real-world tasks such as NYU-Depth-v2 [27] and KITTI [28]. High-resolution imagery and a large amount of data result in steep computational costs in training a network for depth estimation. (2) Depth estimation requires rich spatial details, which is critical to clarify ambiguous local features at each pixel. However, existing NAS search spaces typically incorporate a variety of convolution operations [29,30], ignoring the importance of spatial positional information, which is essential to capturing object structures in monocular depth estimation tasks [31]. Moreover, an image may involve complex scenes, as shown in Fig. 1. Some objects can have a very large spatial representation, while others have significantly smaller representations. As a result, using a single type of convolutional operator for such complex images is not optimal. Recently, multiscale representation learning enables the analysis of images or feature maps at various scales, allowing for the capture of different levels of detail about the content of a scene. Existing multiscale representation learning methods [32] combine multiscale features from various types of the convolutional kernel to improve the model performance for many computer vision tasks. Concurrently, NAS usually addresses this problem by integrating multiple types of convolution with different kernel sizes in the search space [33]. Nevertheless, large convolutional kernel size inevitably increases the model and computational complexity.

To tackle the aforementioned challenges, we draw inspiration from the technique of one-shot NAS [34] to develop an efficient evolutionary neural architecture search framework, called Mde-EvoNAS, which enables automatic encoder design for monocular depth estimation. The key contributions are summarized below:

- We propose an efficient evolutionary neural architecture search framework for monocular depth estimation termed Mde-EvoNAS. The Mde-EvoNAS constructs the search space as a supernet and stochastic train the supernet based on the parent population. Then, offspring individuals can directly reuse the weights of the supernet, and none of them needs to be trained at all to evaluate their fitness value. This way, the supernet and architecture parameters are jointly optimized during the evolutionary search, efficiently improving the training performance of the supernet while still being able to reduce the search budget.
- In response to the lack of spatial positional information in the NAS search space, the multiscale spatial feature awareness (MSFA) module is developed and embedded into the different parts of the search space. The MSFA module further captures the objects of interest by position-sensitive information from different scales of feature maps. Experiments demonstrate that the proposed module can effectively reduce depth estimation error.
- We demonstrate that Mde-EvoNAS leads to highly competitive performance on monocular depth estimation, achieving higher accuracy on KITTI, NYU-Depth-v2, and the intestinal tissue depth estimation task than state-of-the-art manually designed architectures.

2. Related work

In this section, we introduce a brief overview of the recent advances in monocular depth estimation and neural architecture search methods.

2.1. Deep learning-based monocular depth estimation methods

Monocular depth estimation refers to the process of estimating the depth information of each pixel in a scene using a single image captured by one camera. Recently, deep learning methods have significantly advanced monocular depth estimation. These methods primarily rely on deep neural network architectures for feature extraction and depth estimation.

Yang et al. [35] propose the GEDepth module based on ground embedding to decouple pictorial cues and camera parameters to future enhance generalizability for monocular depth estimation. Godard et al. [36] develop auto-masking loss and a minimum reprojection loss to handle occlusions and ignore outlier pixels. Li et al. [37] adopt a

transformer decoder for adaptive bin generation and multiscale decoding to enhance spatial geometry understanding, achieving superior performance on KITTI, NYU, and SUN RGB-D datasets. Fu et al. [7] employ a spacing-increasing discretization strategy and a multiscale network structure to achieve higher accuracy and faster convergence. Shao et al. [38] introduce the Iterative Elastic Bins (IEBins) method for monocular depth estimation, refines depth prediction through multiple stages of progressively finer-grained searches, utilizing elastic target bins to adjust for depth uncertainty, and leverages a GRU-based framework for enhanced temporal context modeling.

However, these methods primarily focus on developing more effective loss functions and sophisticated decoder architectures. Few have considered the impact of the encoder architecture in monocular depth estimation tasks. Med-EvoNAS is complementary to existing methods and can be integrated as a plug-in algorithm to enhance current research outcomes.

2.2. Evolutionary algorithm

Evolutionary algorithms excel in solving large-scale complex problems by efficiently exploring the solution space through mechanisms inspired by natural selection, allowing them to find good solutions to problems with many variables and constraints. Nguyen-Ngoc et al. [39] proposed a hybrid approach that combines deep neural networks with an evolutionary algorithm for accurate damage localization and quantification in truss bridges, enhancing SHM efficiency. In the area of functionally graded materials, Tran et al. [40] developed a hybrid method combining higher-order shear deformation theory and modified couple stress theory with BCMO-ANN, effectively optimizing vibration and buckling performance while addressing material property uncertainties. Additionally, Li et al. [41] integrated surrogate models with evolutionary algorithms to optimize structural parameters, improving both computational efficiency and accuracy in structural optimization. Bai et al. [42] introduced the Blood-Sucking Leech Optimizer (BSLO), a bio-inspired optimization technique based on the blood-sucking behavior of leeches, which efficiently solves complex global optimization problems. Lastly, Qin et al. [43] proposed a vibration-based method for the condition assessment of concrete structures, utilizing advanced artificial intelligence and optimization algorithms to analyze vibration data for precise damage detection and health monitoring.

2.3. Neural architecture search

NAS has recently succeeded in automatically customizing networks tailored to specific tasks. The early development of NAS primarily focused on the core task in computer vision—image classification. Recent studies have reported remarkable empirical results that exceed human-level performance across various image classification benchmarks
[9,19,29,44,45].

Many recent efforts have focused on extending NAS methods to higher-level tasks beyond classification. DetNAS [12] and FNA++ [13] use NAS to demonstrate the importance of backbone architecture design for object detection tasks. Lu et al. [46] propose CitySeg/MOP benchmark test suite integrated into EvoXBench [47], transforms Hardware-aware Neural Architecture Search (HW-NAS) for real-time semantic segmentation into standard multi-objective optimization problems, facilitating comprehensive evaluation and improvement of algorithms in autonomous driving scenarios. Yang et al. [48] enhance Transformer-based knowledge tracing (KT) by incorporating convolution operations for better local context modeling behavior and introduce an evolutionary neural architecture search approach for automated input feature selection and optimal architecture determination, demonstrating effectiveness on challenging real-world education datasets. Cheng et al. [49] introduce the first end-to-end hierarchical NAS framework tailored for deep stereo matching, leveraging task-specific human knowledge to optimize the entire search process jointly. Saikia et al. [50] adopt GD-based NAS methods and Bayesian optimization to efficiently optimize large-scale encoder–decoder neural architectures. The optimized networks achieve state-of-the-art performance in disparity estimation without requiring extensive computational resources. Zhang et al. [51] address the challenges of deep stereo models in continuously learning and adapting to new scenes while maintaining performance on previously seen scenes by introducing the Reusable Architecture Growth (RAG) framework. Huynh et al. [52] introduce LiDNAS, a novel neural architecture search method using Assisted Tabu Search for efficient exploration, producing lightweight monocular depth estimation models that outperform state-of-the-art approaches in efficiency and performance on dense estimation datasets while being significantly more compact.

3. Methodology

In this section, we first introduce the architecture of the proposed search space and describe the MSFA module in detail. Then, we present the proposed algorithm.

3.1. Search space design

Search space refers to the set of all possible network architectures that an algorithm can explore when searching for the optimal neural network structure. The definition of the search space is one of the critical factors for the success of NAS, as it directly impacts the performance of the final model and the efficiency of the search process. The guidelines for designing modern CNNs typically include two main aspects: network-level and cell-level [33]. The network-level design consists of the determination of the width (the number of channels), depth (the number of layers), and spatial resolution change (the number of pooling layers). Previous works, e.g., ResNet, DenseNet, and its variances, are often designed by stacking several blocks to construct a complete network. The cell-level design usually specifies layer-wise connections and computations. Therefore, we follow the same network-level and cell-level design methodology as in [18,33,53,54].

To be specific, a cell is a small convolutional module that is typically repeated multiple times to construct the entire monocular depth estimation model. To ensure scalability across feature maps with varying resolutions, two types of cells are used to process intermediate feature information: (1) the normal cell, which preserves the spatial resolution of the input, and (2) the reduction cell, which halves the spatial resolution by applying a stride of two. Each block within a cell functions as a computational node that has two inputs and one output. Both types of cells are constructed using directed acyclic graphs (DAGs) that consist of five computational blocks. These blocks follow a binary tree structure and end with an elementwise addition operation. As shown in Fig. 2, the construction of a block involves several key design decisions, including input selection and computation operations. Inputs for each block are selected from one of three possible sources: the output of the previous cell h_{i-1}, the output of the cell before that h_{i-2}, or intermediate hidden states generated within the current cell i. The computation operations are chosen from a predefined set of options, which include identity mapping, 3×3 average pooling, 3×3 max pooling, 3×3 depthwise separable convolution, and 5×5 depthwise separable convolution.

3.2. Multiscale spatial feature awareness module design

In depth estimation tasks, channel information helps the depth estimation model extract and integrate various features of the image. In contrast, positional information ensures the model can accurately understand and reconstruct the spatial structure of the image. Combining these two elements is crucial for achieving high-quality depth

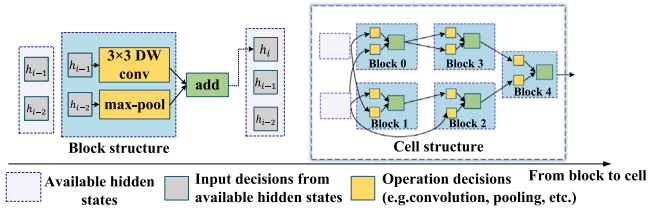


Fig. 2. Detailed structure of the cell.

estimation results. As demonstrated in [55], the standard convolution neglects the importance of channel relationships. In addition, different objects may have different representations in the image.

To effectively capture the wide range of objects and their varying scales, relying on a single type of convolutional kernel with a fixed spatial size may not be the most effective approach for handling this complexity. Our design refers to the coordinate attention mechanism [56]. We propose a multiscale spatial feature awareness (MSFA) module to process the input at multiple scales and capture position information based on multiscale feature maps to augment the feature representations of the interested objects at different scales. In the following, we will describe it in detail.

The structure of the MSFA module is depicted in Fig. 4. Given the input X, a three-branch transformation F_{b-tr} mapping an input $X \in \mathbb{R}^{H\times W\times C}$ to multiscale feature maps $X' \in \mathbb{R}^{H\times W\times C}$ is a three-branch computational unit. Different branches use different types of convolutional kernels. To reduce the overhead model complexity, we utilize grouped convolution at each branch. Specifically, the input is split into different groups (g), and the convolutional kernels (k) are independently applied for each group. The output feature maps in each branch are concatenated. The number of input and output feature maps of the F_{b-tr} should be equal. This step aims to capture different levels of details about the scene's content, which can be formulated as follows:

$$X' = \text{Concat}\left(\sum_{i=1}^{3} \text{GroupConv}_{k_i\times k_i,\,\text{group}=g_i}^{C_i}(X)\right),$$
$$\text{where } k_i \in \{3,5,7\},\ g_i \in \{1,4,8\},\ C_i \in \{\frac{C}{4},\frac{C}{4},\frac{C}{2}\} \tag{1}$$

Where k_i refers to convolutional kernel size, g_i is the number of groups, c_i is the output channels of each branch.

Next, the multiscale feature maps X' are independently processed by 1D pooling kernels $(1, W)$ and $(H, 1)$ along the vertical coordinate and horizontal coordinate, respectively, which can be formulated as follows:

$$X_h = \frac{1}{W}\sum_{i=0}^{W-1} X'(h,i), \quad X_w = \frac{1}{H}\sum_{j=0}^{H-1} X'(j,w) \tag{2}$$

Sequentially, based on the aggregated feature maps produced by Eq. (2), we first concatenate them and then send them to a convolutional transformation function with three components: 1*1 convolutional layer, Batch Normalization, and ReLU, yielding

$$Z = \text{ReLU}(\text{BN}(\text{Conv}_{1\times1}(\text{Concat}(X_h, X_w)))) \tag{3}$$

Where $Z \in \mathbb{R}^{\frac{C}{r}\times(H\times W)}$ is the intermediate feature maps that contains spatial information, and r is the reduction ratio, as in the [56].

The final step aims to capture the coordinate attention based on multiscale feature maps, which can be formulated as follows:

$$Y = X \cdot \sigma(\text{Conv}_{1\times1}(Z[:,:H,:,:])) \cdot \sigma(\text{Conv}_{1\times1}(Z[:,H:,:,:])) \tag{4}$$

Here, σ is the sigmoid function, and Y is the output of our MSFA module.

3.3. Mde-EvoNAS pipeline

The evolutionary algorithm is one of the commonly used search strategies in NAS. Despite their strong performance in various optimization tasks [57,58], evolutionary algorithms (EAs) typically face high computational costs due to their nature as population-based optimization methods. This is especially the case for evolutionary-based NAS methods, as EAs generally require numerous fitness evaluations for each generation. Each fitness evaluation in NAS is computationally costly because it typically involves training a deep neural network from scratch on extensive data.

To solve the above problem, we construct the search space as a supernet based on the technique of one-shot NAS [59] and develop the corresponding evolutionary operators to improve the search efficiency using the weight-sharing technique. For some one-shot methods [18, 59], the discrete search space is relaxed into a continuous one, leading to a deep coupling of the weights among different sub-networks. In Mde-EvoNAS, each individual is a single path in the supernet. We developed a path-wise supernet training approach based on the working mechanisms of EAs to ensure that the trained supernet accurately reflects the relative performance of the candidate networks. Specifically, Mde-EvoNAS randomly chooses an individual from the parent population at each generation and trains it on this mini-batch data. The above process repeats until each epoch's maximum number of iterations is reached. Hence, at each mini-batch training, only one single path is chosen for feedforward and backward propagation. No gradient or weight update acts on other paths in the supernet. Consequently, offspring individuals can directly reuse the weights of the supernet, and none of them needs to be trained at all to evaluate their fitness value. As a result, Mde-EvoNAS effectively reduces the high computational costs typically required for the evolutionary-based NAS method, making it possible to perform monocular depth estimation tasks.

The main components of the Mde-EvoNAS are illustrated in Fig. 3. At first, an initial population P_0 consisting of N_p randomly generated individuals. Each individual in the Mde-EvoNAS represents a monocular depth estimation model. Generally, the architecture of a monocular depth estimation model is constructed by encoder and decoder. In this work, the encoder part is designed by the Mde-EvoNAS, and the decoder part adopts the Laplacian pyramid architecture [60]. As is shown in Fig. 4, the encoder structure is constructed based on normal cell, reduction cell, and the MSFA module.

Algorithm 1: Framework of Mde-EvoNAS

Input: The population size N_p, the maximal epoch number T.
Output: The discovered best architecture of encoder.

```
P_0 ← Initialize population with the size of N_p;
t ← 0;
while t < T do
    for each batch do
        Sample random child network from population P_t;
        Forward pass;
        Backward pass;
    end
    Q_t ← Generate offspring from selected parent individuals
      using the crossover and mutation operators;
    Evaluate the fitness of all individuals in Q_t on a subset of
      dataset_val;
    P_{t+1} ← Environment selection from P_t ∪ Q_t;
    Select the best individual α_best from P_{t+1} and evaluate it on
      the whole dataset_val;
    t ← t + 1;
end
return The individual α_best having the best fitness in P_t.
```

Then, we repeat the following steps. Each individual in P_t is randomly chosen and trained on the mini-batch data until each epoch's

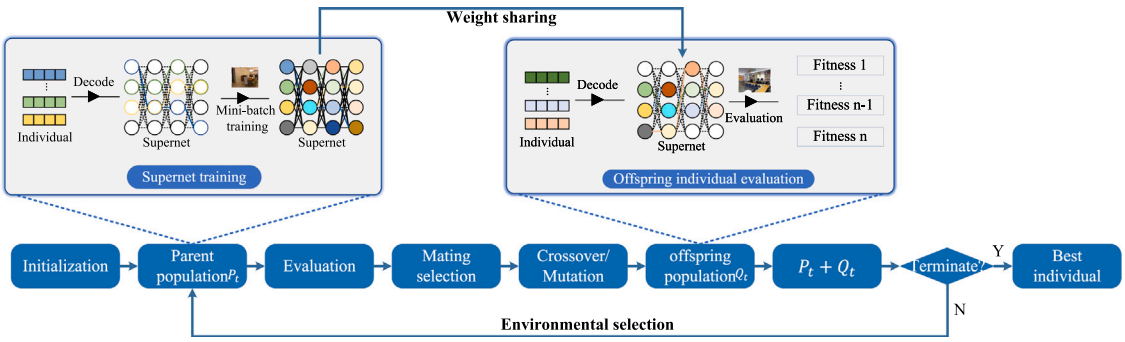


Fig. 3. Overall framework of Mde-EvoNAS.

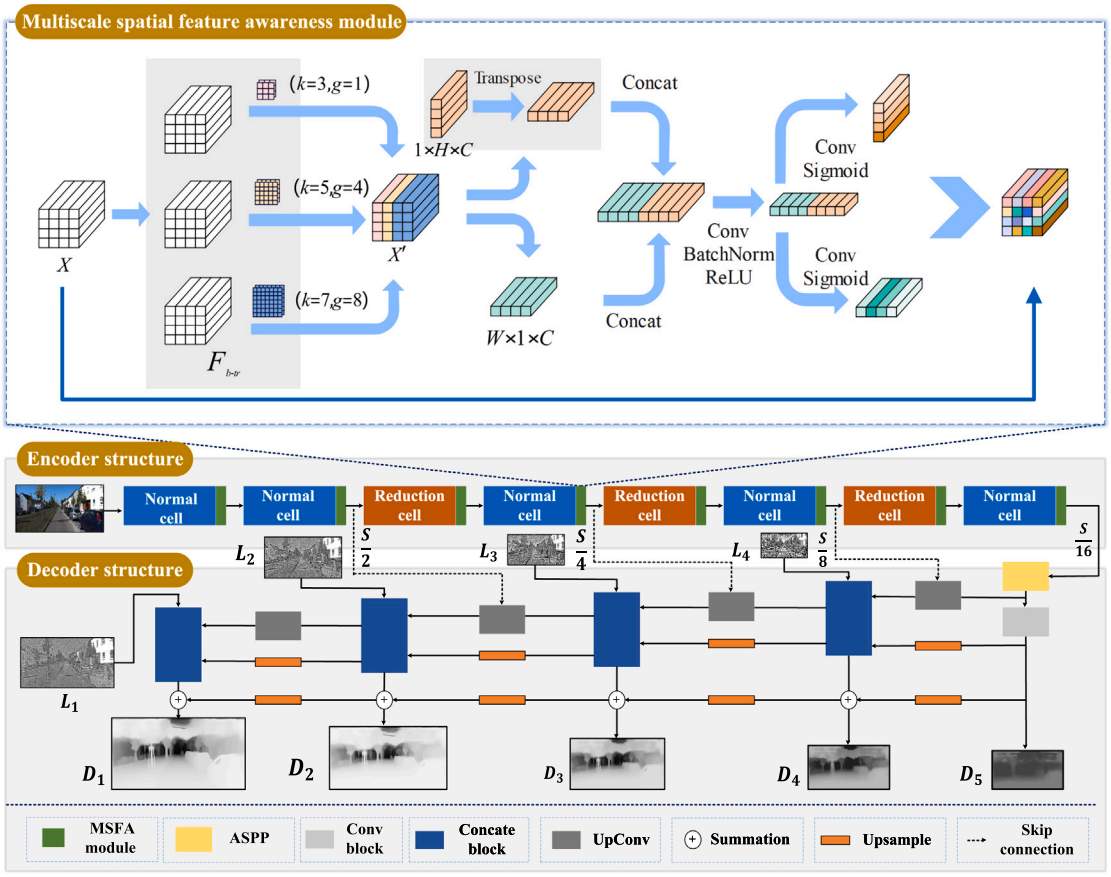


Fig. 4. Example of the monocular depth estimation model designed by Mde-EvoNAS.

maximum number of iterations is reached. Subsequently, the individuals in P_t are evaluated based on trained supernet. Binary tournament selection is used to choose two parent individuals who undergo crossover and mutation to produce two offspring. This process is repeated until N_p offspring are generated, forming the offspring population Q_t. All individuals of Q_t are directly evaluated by activating the corresponding nodes of the trained supernet. In the environmental selection, P_t and Q_t are merged to form a combined population, from which the best N_p individuals are chosen as the parent population of the next generation P_{t+1}. Finally, Mde-EvoNAS outputs the best individual and decodes it into the corresponding monocular depth estimation model.

Algorithm 1 lists the main components of the Mde-EvoNAS. Note that the proposed Mde-EvoNAS jointly optimizes the supernet and architecture parameters during the evolutionary search. The offspring individuals can share the weight of the trained supernet and do not need to be trained for fitness evaluations. This way, Mde-EvoNAS can efficiently improve the training performance while reducing the search budget.

4. Experimental settings

To demonstrate efficiency and robustness, the performance of Mde-EvoNAS is evaluated using two benchmark datasets. In this section, we first propose the benchmark datasets in this work, followed by the implementation details.

4.1. Benchmark datasets

KITTI: The KITTI dataset [28] is an outdoor dataset that consists of various road environments acquired from autonomous driving scenarios. The resolution of images is 375 × 1242 pixels. For the performance comparison, we adopt Eigen training/testing split [61]. This scheme

Table 1
Performance comparison on KITTI benchmark.

Network	Param	FLOPs	$\delta_1 \uparrow$	$\delta_2 \uparrow$	$\delta_3 \uparrow$	Abs Rel ↓	Sq Rel ↓	RMSE ↓	$\text{RMSE}_{\log}$ ↓
ResNeXt101 [63]	73.6M	264.9G	0.921	0.984	0.996	0.083	0.379	3.229	0.127
ResNet101 [64]	43.1M	188.6G	0.914	0.982	0.995	0.086	0.421	3.368	0.133
DenseNet161 [65]	33.5M	198.7G	0.922	0.984	0.996	0.081	0.362	3.183	0.125
InceptionV3 [66]	17.7M	97.1G	0.910	0.978	0.996	0.087	0.407	3.323	0.133
EfficientNet-B7 [67]	13.1M	36.4G	0.914	0.983	0.996	0.086	0.407	3.354	0.132
WideResNet101 [68]	98.4M	322.5G	0.922	0.984	0.996	0.082	0.369	3.217	0.127
RegNet [69]	36.8M	131.2G	0.920	0.984	0.996	0.082	0.372	3.224	0.127
ConvNeXt [70]	64.0M	170.8G	0.898	0.978	0.995	0.093	0.451	3.527	0.142
Mde-EvoNAS	17.6M	112.0G	**0.944**	**0.990**	**0.998**	**0.069**	**0.281**	**2.823**	**0.108**

contains 23488 left view images for training and 697 images for testing. We also adopt the central cropping scheme used in [62] for the performance evaluation.

NYU-Depth-v2: The NYU-Depth-v2 dataset is an indoor dataset that includes 120,000 pairs of RGB images and corresponding ground-truth depth maps, captured across 464 different indoor scenes with a resolution of 640 × 480 pixels. We use the same train/test split as previously applied, which consists of 249 scenes for training and 654 images from the remaining 215 scenes for testing, as described in [61]. Due to the lack of perfect synchronization between the RGB images and their corresponding depth maps, we carefully select 36,253 samples from the 249 training scenes. For comparison with existing methods, the predicted depth maps from our proposed approach are center-cropped to 561 × 427 pixels, following the procedure outlined in [28].

4.2. Implementation details

The Mde-EvoNAS is implemented on the PyTorch library [71]. For a fair comparison, all peer competitors use the same decoder architecture as Mde-EvoNAS and all encoder architectures are trained without any pre-trained on ImageNet. Note that, All experiments are performed on two Nvidia GeForce RTX 3090 cards by ourselves. All the parameters in the searched network and peer competitors (i.e., weights of the network) are initialized based on the strategy introduced in [60].

The parameters of Mde-EvoNAS are divided into two parts: (1) evolutionary search and (2) best individual evaluation. During the evolutionary search phase, the population size and the maximal generation number are set to be 20 and 120, respectively. The probabilities of crossover and mutation are set to 0.95 and 0.2, respectively. In evaluating the fitness, each individual is trained by Stochastic Gradient Descent (SGD) with a batchsize of 8 and momentum rate set to 0.9. The weight decay is set to 0.0001. The channel of the first cell is set to 32. The learning rate is initialized to 0.05 with a single period cosine decay as in [17].

Furthermore, we use six metrics introduced by Eigen et al. [61], which have been most widely used for the performance evaluation of monocular depth estimation, defined as follows:

$$\text{Abs Rel} = \frac{1}{|S|}\sum_{y\in S}\frac{|y-y^*|}{y^*}, \tag{5}$$

$$\text{Sq Rel} = \frac{1}{|S|}\sum_{y\in S}\frac{\|y-y^*\|^2}{y^*}, \tag{6}$$

$$\text{RMSE} = \sqrt{\frac{1}{|S|}\sum_{y\in S}\|y-y^*\|^2}, \tag{7}$$

$$\text{RMSE}_{\log} = \sqrt{\frac{1}{|S|}\sum_{y\in S}\|\log y-\log y^*\|^2}, \tag{8}$$

$$\text{Accuracy} = \%\ \text{of}\ y\ \text{s.t.}\ \max\left(\frac{y}{y^*},\frac{y^*}{y}\right) = \delta < thr,$$

$$thr = 1.25, 1.25^2, 1.25^3, \tag{9}$$

$$\log 10 = \frac{1}{|S|}\sum_{y\in S}|\log_{10} y-\log_{10} y^*|. \tag{10}$$

Where y and y^* denote the predicted depth map and the ground truth, respectively. S is the total number of valid pixels in the ground truth. Note that we use threshold accuracy ($\delta < 1.25$) as the fitness value of Mde-EvoNAS. In the best individual evaluation phase, we train the model by SGD with a batchsize of 8 for a maximum number of 60 epochs. Other hyperparameter settings, such as weight decay rate and momentum, are the same as those used in the evolutionary search.

5. Experimental results

In the experiments, we investigate the performance of Mde-EvoNAS in terms of not only the six evaluation metrics but also the model complexity (in terms of parameters), as well as the computational complexity (in terms of FLOPs). Then, we analyze the effectiveness of the proposed MSFA module. Finally, we provide the evolutionary trajectories of Mde-EvoNAS in searching for the best network architecture on the chosen datasets to help the readers know whether the proposed method converges with the adopted parameter settings.

5.1. Results on KITTI and NYU depth v2

Tables 1 and 2 list the quantitative comparison of the performance metrics on the KITTI and NYU depth v2 datasets, respectively. The network designed by the proposed method outperforms previous handcrafted networks for all of the metrics. It is easy to see that our results achieve the best performance without increasing the model complexity and computational complexity. For example, compared with the best-performing manual model (DenseNet161) on the KITTI benchmark, Mde-EvoNAS improves the Abs REL, Sq REL, RMSE, and $\text{RMSE}_{\log}$ by 14.8%, 22.3%, 11.3% and 13.6%, respectively. In comparison, the model and computational complexity decreased by 47.5% and 43.6%, respectively.

Several results of qualitative comparisons with manually designed network architecture are shown in Figs. 5 and 6, respectively. In the case of outdoor environments, the proposed method demonstrated excellent performance in inferring local object details in complex scenarios. As is shown in Fig. 5, most manually designed models fail to estimate the accurate boundary of thin objects, e.g., pedestrians in the first column in Fig. 5. The shapes of pedestrians are often blurry. In comparison, the network designed by Mde-EvoNAS can accurately capture objects with complex and irregular shapes. Compared with peer competitors, the proposed method can preserve sharper edges of the people. Moreover, Mde-EvoNAS reliably provides depth maps with clear depth boundaries in relatively small objects. As shown in the fourth column of Fig. 5, none of the handcrafted models in this experiment could capture the structure of the stick within the dashed box of the input RGB image. In contrast, our model can capture this local detail and correctly infer its depth result. In the case of indoor environments, due to the presence of many objects at a close range, the depth boundaries are closely aligned with the object boundaries, as illustrated in Fig. 6. The interference of dark objects and shadows leads to incorrect predictions in manually designed network architecture. In contrast, the proposed method can identify the edge boards of cabinets in group 2 and the shaded area on the left. Therefore, it is thought

Table 2
Performance comparison on NYU depth v2 benchmark.

Network	Param	FLOPs	$\delta_1 \uparrow$	$\delta_2 \uparrow$	$\delta_3 \uparrow$	Abs Rel ↓	log10 ↓	RMSE ↓	$\text{RMSE}_{\log}$ ↓
ResNeXt101 [63]	73.6M	241.9G	0.791	0.950	0.986	0.161	0.064	0.506	0.189
ResNet101 [64]	43.1M	172.2G	0.776	0.942	0.983	0.164	0.067	0.545	0.199
DenseNet161 [65]	33.5M	181.5G	0.797	0.952	0.986	0.153	0.064	0.515	0.188
InceptionV3 [66]	17.7M	88.7G	0.761	0.937	0.981	0.170	0.070	0.559	0.206
EfficientNet-B7 [67]	13.1M	33.3G	0.751	0.936	0.982	0.170	0.071	0.573	0.208
WideResNet101 [68]	98.4M	294.5G	0.789	0.950	0.986	0.162	0.064	0.511	0.190
RegNet [69]	36.8M	119.8G	0.778	0.948	0.985	0.159	0.067	0.539	0.195
ConvNeXt [70]	64.0M	156.0G	0.727	0.926	0.978	0.186	0.076	0.605	0.221
Mde-EvoNAS	20.5M	133.9G	**0.824**	**0.960**	**0.988**	**0.142**	**0.058**	**0.471**	**0.174**

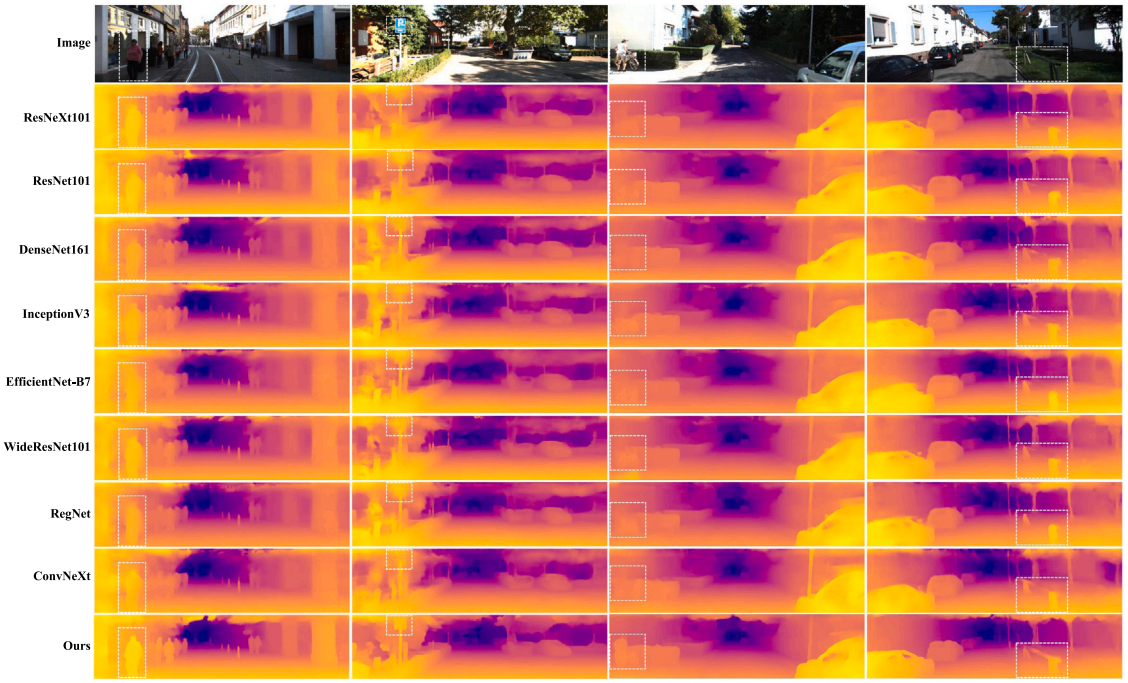


Fig. 5. Qualitative depth comparison on the KITTI dataset. The white boxes indicate the regions to focus on.

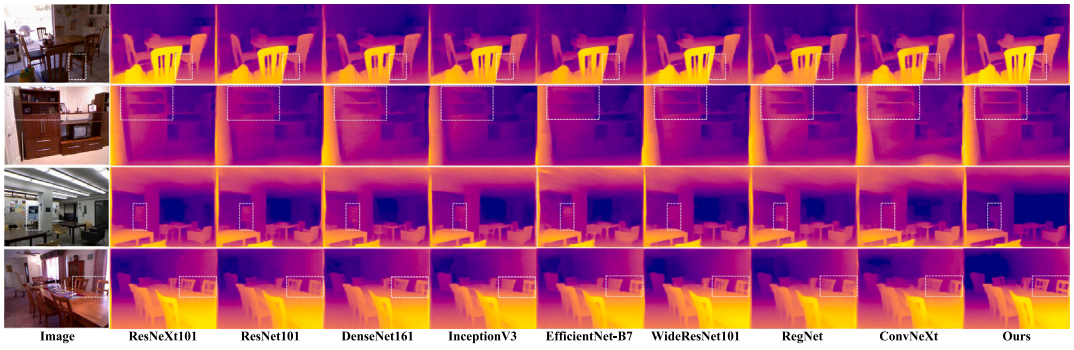


Fig. 6. Qualitative depth comparison on the NYU-Depth-v2 dataset. The white boxes indicate the regions to focus on.

that the model designed by Mde-EvoNAS is effective in accurately estimating the depth information from the color image acquired from various indoor and outdoor environments.

5.2. Comparison with NAS baseline

AE-CNN [26], ENAS [54], and DARTS [19] are effective and popular methods in EvoNAS, RL-based NAS, and GD-based NAS, respectively. In addition, the random search baseline [72] is also competitive. In this work, we include the ENAS, DARTS, and random baseline for comparison to further validate the effectiveness of Mde-EvoNAS, as shown in Table 4 and 5. For a fair comparison, all competitors are performed on the same search space as in Mde-EvoNAS. The parameter settings of the search process are extracted from their seminal papers. The parameter settings of the retraining process are kept the same as described in Mde-EvoNAS. From these results, we observe that the NAS-based methods perform better than Random search baseline. Mde-EvoNAS also surpasses AE-CNN, ENAS and DARTS on all depth benchmark datasets. These experimental results further demonstrate the ability of the proposed method to design high-quality task-dependent models.

Table 3
Comparison with binocular vision model.

Network	Param	FLOPs	Latency(ms)	FPS
PSMNet [73]	5.22M	699.7G	130.4	7.7
Mde-EvoNAS	17.61M	112.0G	**49.9**	**20.1**

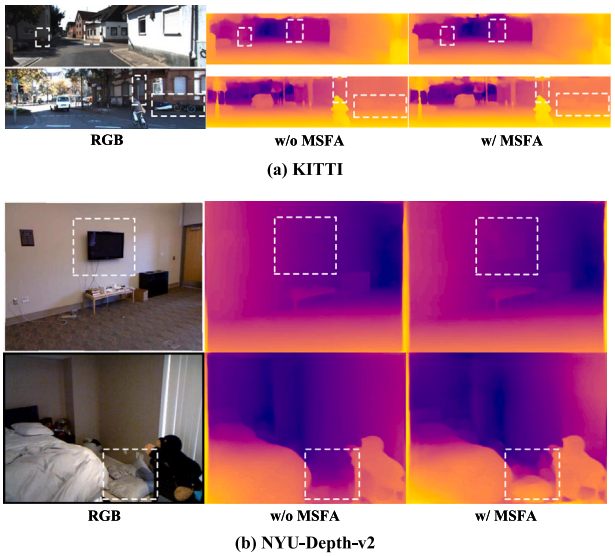


Fig. 7. Qualitative comparisons between Mde-EvoNAS with and without MSFA on the KITTI and NYU-Depth-v2 benchmark.

5.3. Comparison with binocular vision model

To demonstrate the efficiency of monocular depth estimation compared to stereo vision, we conducted a comparative evaluation on the KITTI dataset between our Mde-EvoNAS model and the stereo-based PSMNet model [73]. As shown in Table 3, under the same input image resolution, our model achieves lower inference latency and higher FPS. Although our model has a larger number of parameters, stereo depth estimation requires matching two images and constructing a cost volume, which significantly increases memory usage and computational complexity. Consequently, our model is more efficient and better suited for hardware-constrained applications, where computational resources are limited and real-time performance is essential.

5.4. Effectiveness of MSFA module

In this work, the MSFA module provides the ability to parse the input with multiple scales to capture more detailed information. Then, we add positional information to multiscale feature maps to improve the ability to capture objects at different scales. To check the effectiveness of the MSFA module, we compare the performance of network evolved by Mde-EvoNAS without MSFA.

Table 6 presents the performance of the two networks under comparison on the KITTI and NYU depth v2 benchmark. For the quantitative evaluation, Mde-EvoNAS performs better than Mde-EvoNAS without MSFA, as shown in Table 6. For the qualitative comparisons, we can see that the MSFA performs better in capturing complex image features. Compared to Mde-EvoNAS without MSFA, the street light can be captured by Mde-EvoNAS, as shown in Fig. 7(a). In the case of indoor environments, the MSFA module helps models better capture details in shadows, as shown in Fig. 7(b). The experimental results show that the MSFA module can help the model locate the critical regions in the image more accurately by multiscale feature aggregation, thus improving the depth estimation performance.

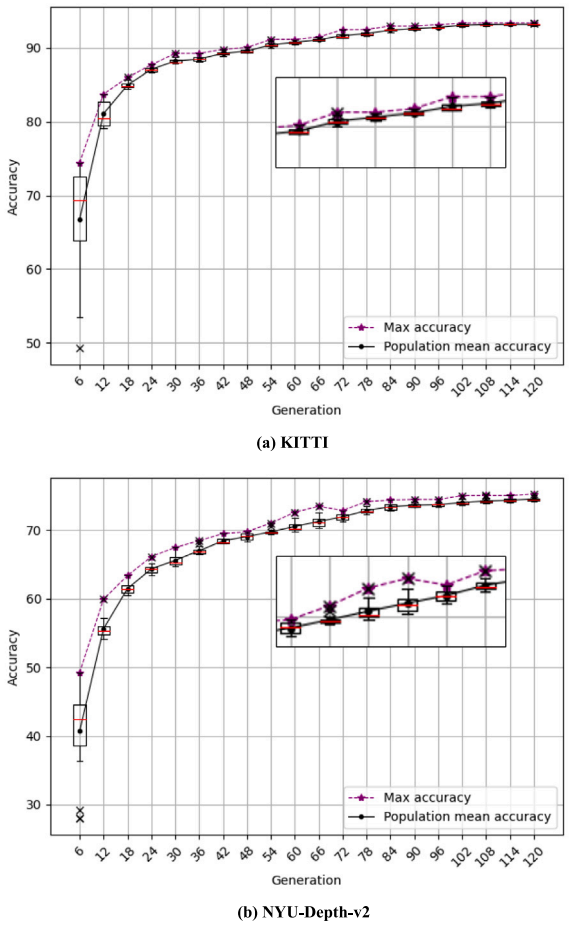


Fig. 8. Evolution trajectories of the Mde-EvoNAS in (a) KITTI and (b) NYU-Depth-v2 benchmark. The horizontal axis denotes the generation number, and the vertical axis denotes the fitness. The black line denotes the mean fitness of the individuals in the same generation, while the purple line denotes the best fitness of the individual in each generation.

5.5. Evolution trajectory

When applying evolutionary algorithms to real-world problems, it is important to determine whether they have converged by the time they finish. A useful method for assessing this is by plotting the evolutionary trajectories. In this section, we present and analyze the evolutionary trajectories of the proposed algorithm based on the benchmark datasets examined, as shown in Fig. 8.

As can be seen from Fig. 8(a), the mean fitness sharply increases from the first generation to the 60 generation and steadily improves as the evolution process proceeds until the 102 generation. In particular, the best and the mean fitness keep the same improvement as the evolutionary process continues. Moreover, the difference between the best fitness and the worst fitness also becomes smaller, which implies that the population converges to a steady state when it terminates. As similar situation can also be seen from Fig. 8(b).

5.6. Transferability analyses

In this section, we validate the transferability performance of the network designed by Mde-EvoNAS on KITTI. We first demonstrate the generality of the network in another monocular depth estimation task. Meanwhile, To further demonstrate the generalization capability of the encoder generated by our method, we directly transfer the encoder architecture to the object detection task and semantic segmentation

Table 4
Performance comparison on KITTI benchmark.

Network	Param	FLOPs	δ_1 ↑	δ_2 ↑	δ_3 ↑	Abs Rel ↓	Sq Rel ↓	RMSE ↓	$RMSE_{log}$ ↓
Random [72]	19.6M	142.9G	0.915	0.983	0.995	0.083	0.399	3.331	0.129
AE-CNN [26]	22.6M	160.2G	0.925	0.987	0.996	0.078	0.365	3.154	0.118
ENAS [54]	17.1M	113.7G	0.927	0.988	0.996	0.079	0.359	3.158	0.122
DARTS [19]	20.4M	131.5G	0.932	0.989	0.996	0.072	0.319	3.017	0.206
Mde-EvoNAS	17.6M	112.0G	**0.944**	**0.990**	**0.998**	**0.069**	**0.281**	**2.823**	**0.108**

Table 5
Performance comparison on NYU depth v2 benchmark.

Network	Param	FLOPs	δ_1 ↑	δ_2 ↑	δ_3 ↑	Abs Rel ↓	Sq Rel ↓	RMSE ↓	$RMSE_{log}$ ↓
Random [72]	23.1M	149.2G	0.780	0.946	0.984	0.160	0.064	0.536	0.193
AE-CNN [26]	19.5M	135.2G	0.796	0.954	0.986	0.145	0.066	0.491	0.179
ENAS [54]	23.5M	153.4G	0.801	0.955	0.986	0.147	0.062	0.495	0.182
DARTS [19]	18.4M	124.2G	0.815	0.956	0.987	0.148	0.061	0.487	0.182
Mde-EvoNAS	20.5M	133.9G	**0.824**	**0.960**	**0.988**	**0.142**	**0.058**	**0.471**	**0.174**

Table 6
Performance comparison of Mde-EvoNAS with and without MSFA on KITTI and NYU depth v2 benchmark.

Network	dataset	δ_1 ↑	δ_2 ↑	δ_3 ↑	Abs Rel ↓	Sq Rel ↓	log10 ↓	RMSE ↓	$RMSE_{log}$ ↓
Mde-EvoNAS (w/o MSFA)	KITTI	0.942	0.989	0.997	0.070	0.289	–	**2.812**	0.109
Mde-EvoNAS (w/ MSFA)	KITTI	**0.944**	**0.990**	**0.998**	**0.069**	**0.281**	–	2.823	**0.108**
Mde-EvoNAS (w/o MSFA)	NYU	0.820	0.959	0.986	**0.140**	–	0.059	0.486	0.177
Mde-EvoNAS (w/ MSFA)	NYU	**0.824**	**0.960**	**0.988**	0.142	–	**0.058**	**0.471**	**0.174**

Table 7
Performance comparison of on Argoverse1.1 benchmark.

Network	Param	δ_1 ↑	Abs Rel ↓	Sq Rel ↓
DenseNet [65]	33.5M	0.723	0.217	2.970
EfficientNet [67]	13.1M	0.728	0.216	2.879
RegNet [69]	36.8M	0.729	0.217	2.960
Mde-EvoNAS	17.6M	**0.755**	**0.196**	**2.648**

Table 8
Performance comparison on MS-COCO benchmark.

Backbone	mAP	mAP0.5	mAP0.75
EfficientNet [67]	0.406	0.565	0.438
DenseNet [65]	0.415	0.575	0.451
RegNet [69]	0.423	0.590	0.459
Mde-EvoNAS	**0.437**	**0.606**	**0.472**

Table 9
Performance comparison on Cityscapes benchmark.

Backbone	$mIoU_{512*1024}$	$mIoU_{1024*1024}$
EfficientNet [67]	0.683	0.702
DenseNet [65]	0.651	0.669
RegNet [69]	0.679	0.703
Mde-EvoNAS	**0.690**	**0.709**

Table 10
The pruning comparison on KITTI dataset.

Method	Pr. rate	Params	Com. rate	δ_1 ↑
No Prune	–	17.61M	1.0×	**0.944**
Direct Prune	90%	0.92M	19.14×	0.905
	99%	0.15M	117.40×	0.866
Prune in NAS	90%	1.31M	16.16×	0.916
	99%	**0.11M**	**146.91**×	0.867

task, respectively. For comparison, we select the DenseNet [65], EfficientNet [67], and RegNet [69] as the baseline. We note that all baseline methods are performed by ourselves using the code released by the authors. The baseline networks also adopt these same experimental settings as the one in Mde-EvoNAS.

5.6.1. Results on Argoverse1.1

Argoverse1.1 [74] is a benchmark dataset specifically designed for autonomous vehicle perception, including 3D tracking, forecasting, and stereo depth estimation, providing diverse and realistic urban driving scenarios. In this work, we directly transfer the network search by KITTI to Argoverse1.1 task. The experimental settings and hyperparameters are the same as the ones adopted in KITTI. Table 7 shows the performance obtained by these networks in the Argoverse1.1 test set. From these results, we observe that the network designed by Mde-EvoNAS has the highest accuracy. The results demonstrate that the network discovered by Mde-EvoNAS can achieve superior generalization performance.

5.6.2. Results on MS-COCO and Cityscapes

Here, we examine the optimized encoder architecture on KITTI, which can be directly used to effectively learn other computer vision tasks, such as MS-COCO [75] and Cityscapes [76]. MS-COCO (Microsoft Common Objects in Context) is a large-scale dataset containing images with diverse objects in complex environments, annotated for detection, segmentation, and keypoint estimation tasks. Cityscapes is a high-quality dataset that focuses on pixel-level annotations of street scenes, primarily used for evaluating semantic segmentation and scene understanding in driving scenarios.

For MS-COCO task, we directly transfer the encoder architecture discovered by Mde-EvoNAS to YOLOV10 [77] object detection framework. For cityscapes, we also transfer the encoder architecture discovered by Mde-EvoNAS to feedformer [78] semantic segmentation framework. In addition, we follow the training setting of YOLOV10 and feedformer, respectively. The comparative results with other baseline networks on the MS-COCO and Cityscapes are summarized in Table 8 and 9, respectively. Specifically, we can see that the encoder searched on KITTI is able to achieve an mAP of 0.437 on MS-COCO and mIOU of 0.690 on Cityscapes, respectively. It can be concluded that the encoder designed on KITTI by Mde-EvoNAS is indeed transferable to other computer vision tasks while maintaining its superiority.

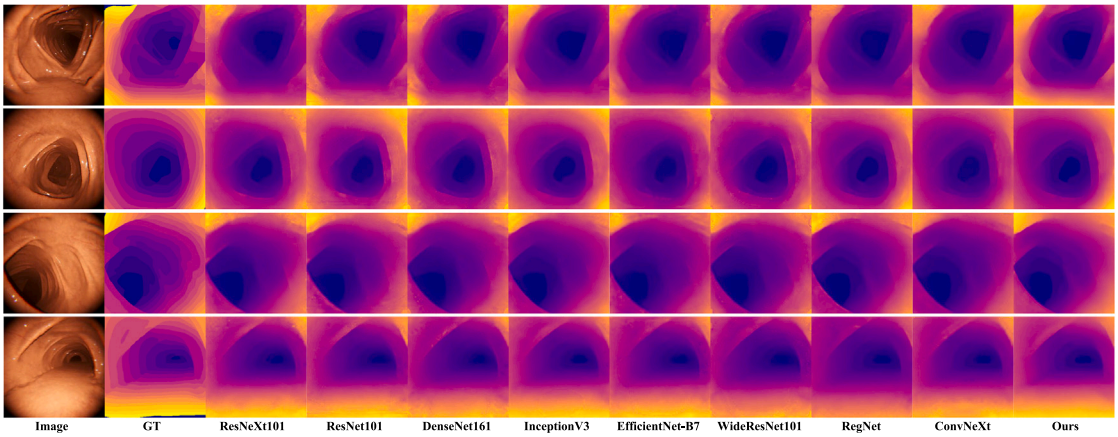


Fig. 9. Qualitative comparison between Mde-EvoNAS with manually designed networks on VR-Caps intestine dataset.

Table 11
Performance comparison on the intestine dataset.

Dataset	Network	Param	$\delta_{0.5}$ ↑	δ_1 ↑	δ_2 ↑	Abs Rel ↓	Sq Rel ↓	RMSE ↓	$\text{RMSE}_{\log}$ ↓
VR-Caps	ResNeXt101 [63]	73.6M	0.845	0.977	0.997	0.064	0.052	0.679	0.090
	ResNet101 [64]	43.1M	0.854	0.977	0.997	0.064	0.050	0.664	0.088
	DenseNet161 [65]	33.5M	0.832	0.973	0.996	0.067	0.055	0.704	0.093
	InceptionV3 [66]	17.7M	0.913	0.986	**0.998**	0.052	0.034	0.527	0.074
	EfficientNet-B7 [67]	13.1M	0.865	0.977	0.997	0.062	0.047	0.631	0.086
	WideResNet101 [68]	98.4M	0.812	0.969	0.996	0.070	0.061	0.737	0.097
	RegNet [69]	36.8M	0.929	0.988	**0.998**	0.048	0.030	0.505	0.070
	ConvNeXt [70]	64.0M	0.907	0.983	**0.998**	0.054	0.038	0.577	0.077
	Mde-EvoNAS	17.6M	**0.948**	**0.989**	**0.998**	**0.043**	**0.024**	**0.443**	**0.064**
Colonscopy	DenseNet161 [65]	33.5M	–	0.943	0.983	0.079	0.027	**0.257**	0.103
	EfficientNet-B7 [67]	13.1M	–	0.922	0.977	0.094	0.037	0.308	0.121
	RegNet [69]	36.8M	–	0.925	0.978	0.088	0.038	0.317	0.121
	Mde-EvoNAS	17.6M	–	**0.964**	**0.996**	**0.067**	**0.022**	0.276	**0.091**

5.7. Network pruning for Mde-EvoNAS

Network pruning optimizes deep neural networks by removing redundant parameters, reducing model size and computational complexity. This improves efficiency, enabling faster inference, lower energy consumption, and easier deployment on resource-constrained devices. Unlike traditional NAS methods for dense networks, DASS [79] introduces a sparsity-aware search space and a reformulated search objective to jointly optimize accuracy and sparsity, allowing architectures to perform well even under extreme pruning ratios.

Inspired by the DASS method, we integrated pruning optimization into our model by replacing its convolutional and linear layers with SparseConv and SparseLinear layers, which include pruning parameters. During the search phase, we alternately train the supernet weights and the pruning parameters in each epoch. The pruning parameters are updated based on the gradient of the loss, effectively incorporating the pruning mechanism into the architecture search process. As a result, the searched optimal architecture is accompanied by its corresponding optimal pruning parameters. In the retraining phase, we fix the optimal pruning parameters and train the searched architecture from scratch.

Table 10 summarizes our results with pruning rates of 90% and 99%, achieving compression ratios of 16× and 146×, respectively, while maintaining strong performance. Additionally, we conducted a comparative experiment by directly pruning the previously searched optimal architecture. Under the extreme 99% pruning rate, models pruned during the search achieved smaller parameter sizes and higher performance than those pruned afterward. This highlights the effectiveness of integrating pruning into the architecture search process for achieving superior trade-offs between model size and performance.

6. Application to intestinal tissue depth estimation

The significance of NAS lies in its ability to automatically design the most optimal network architecture, thereby enhancing model performance and reducing design complexity. NAS algorithms adjust network architectures based on specific tasks and data, optimizing model performance while minimizing the time and effort required for manual design. Hence, NAS allows researchers to design a network architecture with promising performance for a given task without expertise, which is extremely useful for interested users without professional knowledge. In a typical surgical scenario, the doctor usually relies on a limited one-way intestine endoscopy to check the patient's intestinal health. Accurate and dense depth estimation during endoscopy is crucial for doctors to assess the shape and 3D location of the intestinal tissues. This capability significantly impacts human–robot interaction, particularly in guiding the movement of surgical robots and ensuring precise probe operation during procedures. To check the effectiveness of the proposed method, we perform Mde-EvoNAS on a VR-Caps intestine dataset [31] and a synthetic colonoscopy dataset [80] for intestinal tissue depth estimation.

Table 11 compares the performance of Mde-EvoNAS with peer methods extended from existing manually designed networks. The network designed by the proposed method outperforms previous handcrafted networks for all of the metrics. For example, the performance of Mde-EvoNAS exceeds RegNet by a large margin of nearly 1.9 threshold accuracy ($\delta < 0.5$), while the model complexity decreased by 52.1%. From the qualitative results in Fig. 9, the depth maps from handcrafted networks have more blur edges than Mde-EvoNAS. In the examples shown in Fig. 9, Mde-EvoNAS produces more accurate depth maps close to GT with clear edges and details on VR-Caps intestine dataset.

Furthermore, our model achieves the best performance metrics on the synthetic colonoscopy dataset. These results further validate the ability of the Mde-EvoNAS to design task-specific network architectures automatically.

7. Conclusion

This paper introduces Mde-EvoNAS, an efficient evolutionary neural architecture search framework, to construct high-performance monocular depth estimation architectures. In each generation of the evolutionary process, we train the supernet via random sampling of parent individuals. Next, the offspring individuals can inherit the weights from the supernet for fitness evaluation without the need for training. Hence, the computational costs of the fitness evaluations are dramatically reduced. In addition, the proposed MSFA module inherits the advantage of coordinate attention and meanwhile augments the feature representations of objects at different scales. Experiments in KITTI, NYU depth v2, and VR-Caps intestine datasets demonstrate the effectiveness of the proposed method.

Although Mde-EvoNAS can efficiently search a high-performance neural network architecture for monocular depth estimation, the weakness it has is, similar to most existing one-shot NAS approaches, that the supernet training paradigm introduces catastrophic forgetting. Thus, one of our future work will be to improve the training paradigm of the supernet is a critical direction to enhance the reliability of architecture evaluation. In addition, more efficient methods, such as scalable encoding and surrogate models, can be incorporated into the evolutionary search to achieve a good balance between compactness and flexibility.

CRediT authorship contribution statement

Zhihao Yu: Writing – original draft. **Haoyu Zhang:** Writing – review & editing. **Ruyu Liu:** Writing – review & editing, Validation, Methodology. **Sheng Dai:** Writing – review & editing, Methodology, Conceptualization. **Xinan Chen:** Writing – review & editing, Supervision, Project administration, Methodology. **Weiguo Sheng:** Writing – review & editing, Supervision, Project administration, Methodology. **Yaochu Jin:** Writing – review & editing, Supervision, Methodology, Formal analysis.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This research was supported by the National Natural Science Foundation of China under Grant 62306097, 62202137 and Zhejiang Provincial Natural Science Foundation of China under Grant LQ24F030010.

Data availability

Data will be made available on request.

References

[1] R.A. Newcombe, D. Fox, S.M. Seitz, Dynamicfusion: Reconstruction and tracking of non-rigid scenes in real-time, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2015, pp. 343–352.

[2] A. Geiger, P. Lenz, R. Urtasun, Are we ready for autonomous driving? the kitti vision benchmark suite, in: 2012 IEEE Conference on Computer Vision and Pattern Recognition, IEEE, 2012, pp. 3354–3361.

[3] H. Laga, L.V. Jospin, F. Boussaid, M. Bennamoun, A survey on deep learning techniques for stereo-based depth estimation, IEEE Trans. Pattern Anal. Mach. Intell. 44 (4) (2020) 1738–1764.

[4] Z. Zhang, Microsoft kinect sensor and its effect, IEEE Multimedia 19 (2) (2012) 4–10.

[5] X. Chen, H. Ma, J. Wan, B. Li, T. Xia, Multi-view 3d object detection network for autonomous driving, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2017, pp. 1907–1915.

[6] R. Ke, J. Lutin, J. Spears, Y. Wang, A cost-effective framework for automated vehicle-pedestrian near-miss detection through onboard monocular vision, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops, 2017, pp. 25–32.

[7] H. Fu, M. Gong, C. Wang, K. Batmanghelich, D. Tao, Deep ordinal regression network for monocular depth estimation, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 2002–2011.

[8] H. Zhang, Y. Jin, K. Hao, Evolutionary search for complete neural network architectures with partial weight sharing, IEEE Trans. Evol. Comput. 26 (5) (2022) 1072–1086.

[9] Z. Lu, G. Sreekumar, E. Goodman, W. Banzhaf, K. Deb, V.N. Boddeti, Neural architecture transfer, IEEE Trans. Pattern Anal. Mach. Intell. 43 (9) (2021) 2971–2989.

[10] P. Liao, X. Wang, Y. Jin, W. Du, MO-EMT-NAS: Multi-objective continuous transfer of architectural knowledge between tasks from different datasets, 2024, arXiv Preprint arXiv:2407.13122.

[11] Y. Xue, C. Chen, A. Słowik, Neural architecture search based on a multi-objective evolutionary algorithm with probability stack, IEEE Trans. Evol. Comput. 27 (4) (2023) 778–786.

[12] Y. Chen, T. Yang, X. Zhang, G. Meng, X. Xiao, J. Sun, Detnas: Backbone search for object detection, Adv. Neural Inf. Process. Syst. 32 (2019).

[13] J. Fang, Y. Sun, Q. Zhang, K. Peng, Y. Li, W. Liu, X. Wang, FNA++: Fast network adaptation via parameter remapping and architecture search, IEEE Trans. Pattern Anal. Mach. Intell. 43 (9) (2020) 2990–3004.

[14] Z. Lu, R. Cheng, S. Huang, H. Zhang, C. Qiu, F. Yang, Surrogate-assisted multiobjective neural architecture search for real-time semantic segmentation, IEEE Trans. Artif. Intell. 4 (6) (2022) 1602–1615.

[15] Z. Wang, K. Su, J. Zhang, H. Jia, Q. Ye, X. Xie, Z. Lu, Multi-agent automated machine learning, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2023, pp. 11960–11969.

[16] Y. Lin, Y. Endo, J. Lee, S. Kamijo, Bandit-NAS: Bandit sampling and training method for neural architecture search, Neurocomputing 597 (2024) 127684, http://dx.doi.org/10.1016/j.neucom.2024.127684, URL https://www.sciencedirect.com/science/article/pii/S0925231224004557.

[17] Z. Zhong, Z. Yang, B. Deng, J. Yan, W. Wu, J. Shao, C.-L. Liu, Blockqnn: Efficient block-wise neural network architecture generation, IEEE Trans. Pattern Anal. Mach. Intell. 43 (7) (2020) 2314–2328.

[18] H. Liu, K. Simonyan, Y. Yang, DARTS: Differentiable architecture search, in: International Conference on Learning Representations.

[19] S. Xue, R. Wang, B. Zhang, T. Wang, G. Guo, D. Doermann, Idarts: Interactive differentiable architecture search, in: Proceedings of the IEEE/CVF International Conference on Computer Vision, 2021, pp. 1163–1172.

[20] L. Huang, S. Sun, J. Zeng, W. Wang, W. Pang, K. Wang, U-DARTS: Uniform-space differentiable architecture search, Inform. Sci. 628 (2023) 339–349, http://dx.doi.org/10.1016/j.ins.2023.01.129, URL https://www.sciencedirect.com/science/article/pii/S002002552300141X.

[21] Y. Shu, W. Wang, S. Cai, Understanding architectures learnt by cell-based neural architecture search, in: International Conference on Learning Representations.

[22] Q. Lin, Z. Liu, Y. Yang, K.-C. Wong, Y. Lu, J. Li, Multi-objective evolutionary neural architecture search for network intrusion detection, Swarm Evol. Comput. 91 (2024) 101702, http://dx.doi.org/10.1016/j.swevo.2024.101702, URL https://www.sciencedirect.com/science/article/pii/S2210650224002402.

[23] T. Gong, Y. Ma, PSO-based lightweight neural architecture search for object detection, Swarm Evol. Comput. 90 (2024) 101684, http://dx.doi.org/10.1016/j.swevo.2024.101684, URL https://www.sciencedirect.com/science/article/pii/S2210650224002220.

[24] Q.M. Phan, N.H. Luong, Parameter-less Pareto local search for multi-objective neural architecture search with the interleaved multi-start scheme, Swarm Evol. Comput. 87 (2024) 101573, http://dx.doi.org/10.1016/j.swevo.2024.101573, URL https://www.sciencedirect.com/science/article/pii/S2210650224001111.

[25] R. Shang, S. Zhu, H. Liu, T. Ma, W. Zhang, J. Feng, L. Jiao, R. Stolkin, Evolutionary architecture search via adaptive parameter control and gene potential contribution, Swarm Evol. Comput. 82 (2023) 101354, http://dx.doi.org/10.1016/j.swevo.2023.101354, URL https://www.sciencedirect.com/science/article/pii/S221065022300127X.
[26] Y. Sun, B. Xue, M. Zhang, G.G. Yen, J. Lv, Automatically designing CNN architectures using the genetic algorithm for image classification, IEEE Trans. Cybern. 50 (9) (2020) 3840–3854.
[27] N. Silberman, D. Hoiem, P. Kohli, R. Fergus, Indoor segmentation and support inference from rgbd images, in: Computer Vision–ECCV 2012: 12th European Conference on Computer Vision, Florence, Italy, October 7-13, 2012, Proceedings, Part V 12, Springer, 2012, pp. 746–760.
[28] A. Geiger, P. Lenz, C. Stiller, R. Urtasun, Vision meets robotics: The kitti dataset, Int. J. Robot. Res. 32 (11) (2013) 1231–1237.
[29] Y. Liu, Y. Sun, B. Xue, M. Zhang, G.G. Yen, K.C. Tan, A survey on evolutionary neural architecture search, IEEE Trans. Neural Netw. Learn. Syst. 34 (2) (2021) 550–570.
[30] T. Elsken, J.H. Metzen, F. Hutter, Neural architecture search: A survey, J. Mach. Learn. Res. 20 (55) (2019) 1–21.
[31] R. Liu, Z. Liu, H. Zhang, G. Zhang, Z. Zuo, W. Sheng, Dense depth completion based on multi-scale confidence and self-attention mechanism for intestinal endoscopy, in: 2023 IEEE International Conference on Robotics and Automation, ICRA, IEEE, 2023, pp. 7476–7482.
[32] L. Jiao, M. Wang, X. Liu, L. Li, F. Liu, Z. Feng, S. Yang, B. Hou, Multiscale deep learning for detection and recognition: A comprehensive survey, IEEE Trans. Neural Netw. Learn. Syst. (2024) 1–21, http://dx.doi.org/10.1109/TNNLS.2024.3389454.
[33] Z. Lu, I. Whalen, Y. Dhebar, K. Deb, E.D. Goodman, W. Banzhaf, V.N. Boddeti, Multiobjective evolutionary design of deep convolutional neural networks for image classification, IEEE Trans. Evol. Comput. 25 (2) (2020) 277–291.
[34] X. Chu, S. Lu, X. Li, B. Zhang, Mixpath: A unified approach for one-shot neural architecture search, in: Proceedings of the IEEE/CVF International Conference on Computer Vision, 2023, pp. 5972–5981.
[35] X. Yang, Z. Ma, Z. Ji, Z. Ren, Gedepth: Ground embedding for monocular depth estimation, in: Proceedings of the IEEE/CVF International Conference on Computer Vision, 2023, pp. 12719–12727.
[36] C. Godard, O. Mac Aodha, M. Firman, G.J. Brostow, Digging into self-supervised monocular depth estimation, in: Proceedings of the IEEE/CVF International Conference on Computer Vision, 2019, pp. 3828–3838.
[37] Z. Li, X. Wang, X. Liu, J. Jiang, Binsformer: Revisiting adaptive bins for monocular depth estimation, IEEE Trans. Image Process. (2024).
[38] S. Shao, Z. Pei, X. Wu, Z. Liu, W. Chen, Z. Li, Iebins: Iterative elastic bins for monocular depth estimation, Adv. Neural Inf. Process. Syst. 36 (2024).
[39] L. Nguyen-Ngoc, Q. Nguyen-Huu, G. De Roeck, T. Bui-Tien, M. Abdel-Wahab, Deep neural network and evolved optimization algorithm for damage assessment in a truss bridge, Math. 12 (15) (2024) 2300.
[40] V.-T. Tran, T.-K. Nguyen, H. Nguyen-Xuan, M.A. Wahab, Vibration and buckling optimization of functionally graded porous microplates using BCMO-ANN algorithm, Thin-Walled Struct. 182 (2023) 110267.
[41] Y. Li, H.-L. Minh, M. Cao, X. Qian, M.A. Wahab, An integrated surrogate model-driven and improved termite life cycle optimizer for damage identification in dams, Mech. Syst. Signal Process. 208 (2024) 110986.
[42] J. Bai, H. Nguyen-Xuan, E. Atroshchenko, G. Kosec, L. Wang, M.A. Wahab, Blood-sucking leech optimizer, Adv. Eng. Softw. 195 (2024) 103696.
[43] S. Qin, J. Feng, J. Tang, X. Huo, Y. Zhou, F. Yang, M.A. Wahab, Condition assessment of a concrete filled steel tube arch bridge using in-situ vibration measurements and an Improved Artificial Fish Swarm Algorithm, Comput. Struct. 291 (2024) 107213.
[44] J. Sun, W. Yao, T. Jiang, X. Chen, Efficient search of comprehensively robust neural architectures via multi-fidelity evaluation, Pattern Recognit. 146 (2024) 110038.
[45] S. Liu, H. Zhang, Y. Jin, A survey on computationally efficient neural architecture search, J. Autom. Intell. 1 (1) (2022) 100002.
[46] Y. Zhao, Z. Liang, Z. Lu, R. Cheng, A multi-objective optimization benchmark test suite for real-time semantic segmentation, 2024, arXiv Preprint arXiv:2404.16266.
[47] B. Huang, R. Cheng, Z. Li, Y. Jin, K.C. Tan, EvoX: A distributed GPU-accelerated framework for scalable evolutionary computation, IEEE Trans. Evol. Comput. (2024).
[48] S. Yang, X. Yu, Y. Tian, X. Yan, H. Ma, X. Zhang, Evolutionary neural architecture search for transformer in knowledge tracing, Adv. Neural Inf. Process. Syst. 36 (2024).
[49] X. Cheng, Y. Zhong, M. Harandi, Y. Dai, X. Chang, H. Li, T. Drummond, Z. Ge, Hierarchical neural architecture search for deep stereo matching, Adv. Neural Inf. Process. Syst. 33 (2020) 22158–22169.
[50] T. Saikia, Y. Marrakchi, A. Zela, F. Hutter, T. Brox, Autodispnet: Improving disparity estimation with automl, in: Proceedings of the Ieee/Cvf International Conference on Computer Vision, 2019, pp. 1812–1823.
[51] C. Zhang, K. Tian, B. Fan, G. Meng, Z. Zhang, C. Pan, Continual stereo matching of continuous driving scenes with growing architecture, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2022, pp. 18901–18910.
[52] L. Huynh, P. Nguyen, J. Matas, E. Rahtu, J. Heikkilä, Lightweight monocular depth with a novel neural architecture search method, in: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision, 2022, pp. 3643–3653.
[53] B. Zoph, Q. Le, Neural architecture search with reinforcement learning, in: International Conference on Learning Representations, 2016.
[54] H. Pham, M. Guan, B. Zoph, Q. Le, J. Dean, Efficient neural architecture search via parameters sharing, in: International Conference on Machine Learning, PMLR, 2018, pp. 4095–4104.
[55] J. Hu, L. Shen, G. Sun, Squeeze-and-excitation networks, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 7132–7141.
[56] Q. Hou, D. Zhou, J. Feng, Coordinate attention for efficient mobile network design, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2021, pp. 13713–13722.
[57] Y. Sun, G.G. Yen, Z. Yi, Improved regularity model-based EDA for many-objective optimization, IEEE Trans. Evol. Comput. 22 (5) (2018) 662–678.
[58] Y. Sun, G.G. Yen, Z. Yi, IGD indicator-based evolutionary algorithm for many-objective optimization problems, IEEE Trans. Evol. Comput. 23 (2) (2018) 173–187.
[59] Y. Xu, L. Xie, W. Dai, X. Zhang, X. Chen, G.J. Qi, H. Xiong, Q. Tian, Partially-connected neural architecture search for reduced computational redundancy, IEEE Trans. Pattern Anal. Mach. Intell. 43 (9) (2021) 2953–2970.
[60] M. Song, S. Lim, W. Kim, Monocular depth estimation using laplacian pyramid-based depth residuals, IEEE Trans. Circuits Syst. Video Technol. 31 (11) (2021) 4381–4393.
[61] D. Eigen, C. Puhrsch, R. Fergus, Depth map prediction from a single image using a multi-scale deep network, Adv. Neural Inf. Process. Syst. 27 (2014).
[62] R. Garg, V.K. Bg, G. Carneiro, I. Reid, Unsupervised cnn for single view depth estimation: Geometry to the rescue, in: Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, the Netherlands, October 11-14, 2016, Proceedings, Part VIII 14, Springer, 2016, pp. 740–756.
[63] S. Xie, R. Girshick, P. Dollár, Z. Tu, K. He, Aggregated residual transformations for deep neural networks, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2017, pp. 1492–1500.
[64] K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 770–778.
[65] G. Huang, Z. Liu, L. Van Der Maaten, K.Q. Weinberger, Densely connected convolutional networks, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2017, pp. 4700–4708.
[66] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, Z. Wojna, Rethinking the inception architecture for computer vision, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 2818–2826.
[67] M. Tan, Q.V. Le, EfficientNet: Rethinking model scaling for convolutional neural networks, in: K. Chaudhuri, R. Salakhutdinov (Eds.), Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA, in: Proceedings of Machine Learning Research, vol. 97, PMLR, 2019, pp. 6105–6114, URL http://proceedings.mlr.press/v97/tan19a.html.
[68] S. Zagoruyko, N. Komodakis, Wide residual networks, in: R.C. Wilson, E.R. Hancock, W.A.P. Smith (Eds.), Proceedings of the British Machine Vision Conference 2016, BMVC 2016, York, UK, September 19-22, 2016, BMVA Press, 2016, URL http://www.bmva.org/bmvc/2016/papers/paper087/index.html.
[69] J. Xu, Y. Pan, X. Pan, S. Hoi, Z. Yi, Z. Xu, RegNet: Self-regulated network for image classification, IEEE Trans. Neural Netw. Learn. Syst. 34 (11) (2022) 9562–9567.
[70] Z. Liu, H. Mao, C.-Y. Wu, C. Feichtenhofer, T. Darrell, S. Xie, A convnet for the 2020s, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2022, pp. 11976–11986.
[71] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, A. Lerer, Automatic differentiation in pytorch, 2017.
[72] L. Li, A. Talwalkar, Random search and reproducibility for neural architecture search, in: Uncertainty in Artificial Intelligence, PMLR, 2020, pp. 367–377.
[73] J.R. Chang, Y.S. Chen, Pyramid stereo matching network, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 5410–5418.
[74] M.F. Chang, J. Lambert, P. Sangkloy, J. Singh, S. Bak, A. Hartnett, D. Wang, P. Carr, S. Lucey, D. Ramanan, et al., Argoverse: 3d tracking and forecasting with rich maps, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2019, pp. 8748–8757.

[75] T.Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, C.L. Zitnick, Microsoft coco: Common objects in context, in: Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V 13, Springer, 2014, pp. 740–755.

[76] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, B. Schiele, The cityscapes dataset for semantic urban scene understanding, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 3213–3223.

[77] A. Wang, H. Chen, L. Liu, K. Chen, Z. Lin, J. Han, G. Ding, Yolov10: Real-time end-to-end object detection, 2024, arXiv Preprint arXiv:2405.14458.

[78] J.h. Shim, H. Yu, K. Kong, S.J. Kang, Feedformer: Revisiting transformer decoder for efficient semantic segmentation, in: Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 37, No. 2, 2023, pp. 2263–2271.

[79] H. Mousavi, M. Loni, M. Alibeigi, M. Daneshtalab, DASS: differentiable architecture search for sparse neural networks, ACM Trans. Embed. Comput. Syst. 22 (5s) (2023) 1–21.

[80] A. Rau, P.E. Edwards, O.F. Ahmad, P. Riordan, M. Janatka, L.B. Lovat, D. Stoyanov, Implicit domain adaptation with conditional generative adversarial networks for depth prediction in endoscopy, Int. J. Comput. Assist. Radiol. Surg. (2019) 1–10.