Differentiable Architecture Search With Attention Mechanisms for Generative Adversarial Networks

Yu Xue, *Senior Member, IEEE*, Kun Chen, and Ferrante Neri, *Senior Member, IEEE*

***Abstract*—Generative adversarial networks (GANs) are machine learning algorithms that can efficiently generate data such as images. Although GANs are very popular, their training usually lacks stability, with the generator and discriminator networks failing to converge during the training process. To address this problem and improve the stability of GANs, in this paper, we automate the design of stable GANs architectures through a novel approach: differentiable architecture search with attention mechanisms for generative adversarial networks (DAMGAN). We construct a generator supernet and search for the optimal generator network within it. We propose incorporating two attention mechanisms between each pair of nodes in the supernet. The first attention mechanism, down attention, selects the optimal candidate operation of each edge in the supernet, while the second attention mechanism, up attention, improves the training stability of the supernet and limits the computational cost of the search by selecting the most important feature maps for the following candidate operations. Experimental results show that the architectures searched by our method obtain a state-of-the-art inception score (IS) of 8.99 and a very competitive Fréchet inception distance (FID) of 10.27 on the CIFAR-10 dataset. Competitive results were also obtained on the STL-10 dataset (IS = 10.35, FID = 22.18). Notably, our search time was only 0.09 GPU days.**

***Index Terms*—Generative adversarial networks, neural architecture search, attention mechanism, generative model.**

I. Introduction

GENERATIVE adversarial networks (GANs) [1] are machine learning algorithms that are widely used in computer vision to generate realistic images. Considerable attention has been focused on GANs due to their excellent performance [2], [3], [4]. GANs comprise two neural networks: a generator (G) and a discriminator (D). These two networks train each other with conflicting objectives. Specifically, the G network generates images, and the D network determines whether a generated image is genuine or generated by G. While the objective of G is to generate images that appear genuine to D, the objective of D is to accurately distinguish whether an image is generated or not. Since the generator and discriminator have opposite optimisation goals, the training of GANs is very unstable, which involves repeated oscillation in the networks' parameters without the convergence of the two networks. Researchers have tried a variety of approaches to solve this problem. One approach is related to the definition of loss functions, such as least-square loss [5] and Wasserstein distance [6]. There are also studies to improve the stability of GANs by changing the regularization method [7], [8]. Recently, researchers have found that the architecture of G has an important effect on the training stability of GANs [7], [9], [10]. However, the human-made architectural design of GANs requires expertise and expert experience as well as human judgement.

Neural architecture search (NAS) is a main technique for automatically searching architecture of a deep neural network to accomplish a target task. NAS is successfully used in computer vision tasks. For example, object detection [11], [12], [13], image classification [14], [15], [16], [17], [18], [19], semantic segmentation [20], [21], image prediction [20], [22], [23] and image generation [24], [25], [26], [27], [28], [29]. Some works in NAS employed reinforcement learning and evolutionary algorithms as the search strategy. However, these approaches require the evaluation of a large number of network architectures, and take thousands of GPU days to achieve reasonably good results. To reduce the computational cost, subsequent researchers have proposed various strategies, including weight sharing [30] and performance prediction [31], [32]. Differentiable architecture search (DARTS) [33] makes use of a supernet that includes all possible architectures within a search space, then optimises the weights of the supernet to identify the best architecture. DARTS uses some parameters related to operators to make the discrete search space continuous and to optimise the architectural parameters and network weights by a dual optimisation strategy and the gradient descent technique.

Since DARTS unnecessarily to evaluate a large number of deep neural networks, the computational cost is low and the search is rapid. DARTS selects the operation between two nodes according to the size of architectural parameters. When the supernet converges, the two candidate operations corresponding to the first largest two architectural parameters are selected as the final connecting edges of the two nodes. However, in the initial stage of supernet training, the candidate operations have not

Manuscript received 16 August 2023; revised 9 January 2024; accepted 11 February 2024. Date of publication 21 March 2024; date of current version 24 July 2024. This work was supported in part by the National Natural Science Foundation of China under Grant 62376127, Grant 61876089, Grant 61876185, Grant 61902281, and Grant 61403206, in part by the Natural Science Foundation of Jiangsu Province under Grant BK20141005, in part by the Natural Science Foundation of the Jiangsu Higher Education Institutions of China under Grant 14KJB520025, and in part by the Project of Distinguished Professors of Jiangsu Province. *(Corresponding author: Yu Xue.)*

Yu Xue and Kun Chen are with the School of Software, Nanjing University of Information Science and Technology, Nanjing 210044, China (e-mail: xueyu@nuist.edu.cn; 202212490362@nuist.edu.cn).

Ferrante Neri is with NICE Research Group, Department of Computer Science, University of Surrey, GU2 7XS Guildford, U.K. (e-mail: f.neri@surrey.ac.uk).

Recommended for acceptance by J. Zhou.

Digital Object Identifier 10.1109/TETCI.2024.3369998

been fully trained, which will result in unfair competition among candidate operations and may lead to network collapse [34].

Several recent studies have shown NAS is an effective tool in designing the architecture of GANs [24], [25], [26], [27], [28], [29]. Some researchers have introduced reinforcement learning for designing the architecture of GANs. For example AGAN [24]. In AGAN, recurrent neural networks are trained as controllers to guide the search, and the inception score (IS) is used as a reward function. Although this approach has potential, a computation time of roughly 1,200 GPU days makes it impractical. The study of AutoGAN [25] exploits the algorithmic proposed in [24] and uses parameter sharing and dynamic adjustment to speed up the search process. The modified algorithm, dubbed AutoGAN, greatly reduces the time cost from that reported in [24] but has a relatively small search space. Some other researchers have introduced evolutionary algorithm (EA) for designing the architecture of GANs. For example EAGAN [26]. This approach first training a supernet, and then applying an EA to evolve a population of subnets derived from that supernet. EAGAN uses a sequential evolutionary approach to separate the search of D and G network architectures into two steps. Since the subnets inherit the supernet weight parameters, there is no need for those subnets to be retrained. Thus, EAGAN can efficiently search for architectures with excellent performance. Recently, DARTS based NAS methods are used for designing the architecture of GANs. For example, AdversarialNAS [27] and AlphaGAN [28]. Both of them use a supernet, and subnet sampling is performed via a gradient-based algorithm. The main advantage of this over an EA approach is that a gradient-based algorithm is generally much faster than an EA. AdversarialNAS and AlphaGAN transform the search space from discrete to continuous and adopt a two-level optimisation strategy to optimise the architectural parameters and network weights alternately. Finally, the optimal subnet is determined by the value of the architectural parameters. Inspired by DARTS, AdversarialNAS and AlphaGAN search architectures by alternately optimising network architectural parameters and network weights. However, it has been shown that the DARTS approach is prone to network collapse [35], [36], [37], and the operation corresponding to the largest network architectural weight is not necessarily the most important [38]; inevitably, AdversarialNAS and AlphaGAN also suffer from this problem. Therefore, it is necessary to study a more appropriate method to assess the importance of the candidate operations.

In this paper, we propose a new gradient-based NAS method for GANs: differentiable architecture search with attention mechanisms for generative adversarial networks (DAMGAN). By following the style of DARTS algorithms [27], [28], DAMGAN makes use of a supernet (for G) and extracts a subnet, i.e., a candidate G. However, unlike the algorithms in [27] and [28], DAMGAN does not use the two-level optimisation structure of DARTS. Instead, DAMGAN embeds part of the optimisation process within the supernet structure so that the optimal subnet can be extracted by a single gradient-based optimisation.

The supernet is a network whose nodes contain a large set of feature maps. The nodes are interconnected through multiple (redundant) links, each link representing a candidate operation, such as upsampling or convolution. Between each pair of nodes, two attention mechanisms [39] are inserted. One attention mechanism is placed after the parallel application of all potential candidate operations; it selects the optimal operation to be used in the subnet. This logic replaces the use of architectural parameters to guide search in DARTS, and inspired by the results [38], the attention mechanism can improve the performance of a neural network by focusing on the parts that are most useful and ignoring that have the least impact on the outputs. The DAMGAN method involves accumulating the attention weights of different candidate operations and using the accumulated value as an indicator of the importance of the candidate operations. The insertion of an attention mechanism between each pair of nodes of the supernet causes a dramatic increase in the number of parameters and the computational complexity. To reduce the computational cost of the optimisation, inspired by [34], we propose the insertion of another attention mechanism between each pair of nodes, before the parallel application of candidate operations. This second attention mechanism establishes a ranking criterion for the feature maps outputted by candidate operations, thereby serving as a selection criterion for the most crucial operation among each pair of feature maps. Because the feature maps associated with larger attention weights often have more important features, thus supporting the generation task, this method is able not only to shorten the time cost of the overall search process but also to make the supernet training more stable.

The novel contributions of this study are summarised as follows:

- We propose a novel NAS method for GANs based on DARTS. The proposed method, similar to DARTS, utilises a supernet and samples subnets as candidate architectures. However, DAMGAN modifies the supernet architecture by adding two attention mechanisms between each pair of nodes. The inclusion of these attention mechanisms introduces a unique formulation of the NAS problem, significantly impacting the search space and consequently the representation of candidate architectures. In DAMGAN, attention weights identify an architecture.
- The first attention mechanism selects the most crucial feature maps before and after the candidate operations. Specifically, we apply the attention mechanism to the input feature maps of candidate operations, selecting the most important feature maps as the inputs of candidate operations based on the attention weights. This approach not only diminishes the influence of inferior feature maps on the search process, enhancing stability, but also expedites the search.
- The second attention mechanism is then employed to rank the feature maps outputted by the candidate operations. We use this second attention mechanism to implicitly rank the importance of the candidate operations by assessing the importance of their outputs. This is achieved by applying the attention mechanism to the output feature maps of candidate operations. The sum of the attention weights corresponding to the output feature maps of each operation represents the importance of each operation. This

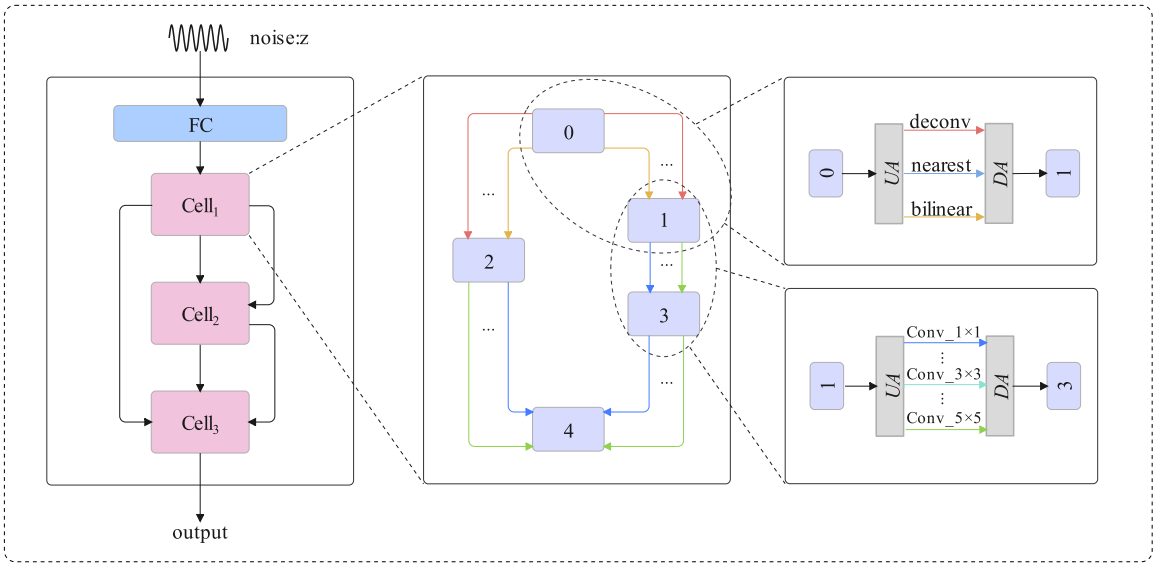


Fig. 1. Generator supernet: entire network (left); cell as interconnected nodes containing feature maps (centre); connections between pairs of nodes, including up and down attention mechanisms, UA and DA, respectively (right).

innovative approach addresses the issue of mismatch between the size of architecture parameters and the importance of candidate operations when conducting differentiable architecture search on GANs.

We conducted extensive experiments on our proposed method using CIFAR-10 and STL-10 datasets, achieving outstanding results within a remarkably limited computational time. On the CIFAR-10 dataset, we achieved new state-of-the-art IS values and competitive FDD values (IS $=$ 8.99, FID $=$ 10.27). We also demonstrated the portability of our searched architectures on the STL-10 dataset. The computation time on the CIFAR-10 dataset was only 0.09 GPU days.

This article is organised in the following way. Section II has presented the proposed DAMGAN, describing the framework and its components. Section III details the experimental setup and reports the numerical results obtained during comparative and ablation studies. Section IV presents the conclusion of this work.

II. Methodology: DAMGAN

In this section, we present our proposed DAMGAN framework. Section II-A describes the overall architecture of the supernet. Sections II-B and II-C focuses on the two novel elements of DAMGAN: the selection of candidate operations and the selection of feature maps, via the novel use of attention mechanisms in the context of NAS. Finally, Section II-D describes the NAS task, i.e., the selection of the optimal subnet from the original supernet using the simple gradient-based training of the supernet.

A. Supernet Architecture

DAMGAN makes use of a supernet describing all potential architectures of the G network. We chose not to design G and D simultaneously as this has been shown to exacerbate the training instability issues [28]. We trained the generator supernet using a network of discriminator that is consistent with the discriminator structure used in AutoGAN [25].

The supernet is represented by the box on the left-hand side of Fig. 1; its input is noise and its output is a generated image. The generator is formed of an initial fully connected layer (FC; in blue in Fig. 1), followed by several cells (in pink in Fig. 1). These cells are interconnected, directly and through skip connections. In this study, we use a representation with three cells, following the model proposed in [28]. Each cell contains several interconnected nodes (five in our example). The details of each cell are shown in the central box of Fig. 1. Each node is identified by a number and contains 256 feature maps. The edges connecting the nodes represent candidate operations. The boxes on the right-hand side of Fig. 1 show the details of two nodes interconnected by candidate operations. We propose the inclusion of two attention mechanisms between each pair of nodes, first $Up\_Attention$ (UA), then $Down\_Attention$ (DA). The role of UA is to assign a weight to each feature map in the departure node; in Fig. 1, in the pair of nodes 1 and 3, node 1 is the departure node. Thus, the feature maps are sorted according to UA based on their importance. The most important feature maps are selected and then undergo all the candidate operations between two nodes. The candidate operations considered in this study are of two types.

- *Upsampling operations:* These operations occur between node 0 and the other nodes. In Fig. 1, these operations apply to two pairs of nodes: $0 \rightarrow 1$ and $0 \rightarrow 2$. We consider three options: 'nearest neighbour sampling', 'bilinear interpolation sampling' and 'deconvolution sampling' (i.e., transposed convolution).
- *Convolution operations:* These operations apply to every pair of nodes from which node 0 is excluded. In Fig. 1, these operations apply to the pairs $1 \rightarrow 3$, $2 \rightarrow 4$ and $3 \rightarrow 4$. We consider six operations: convolution with kernel sizes of

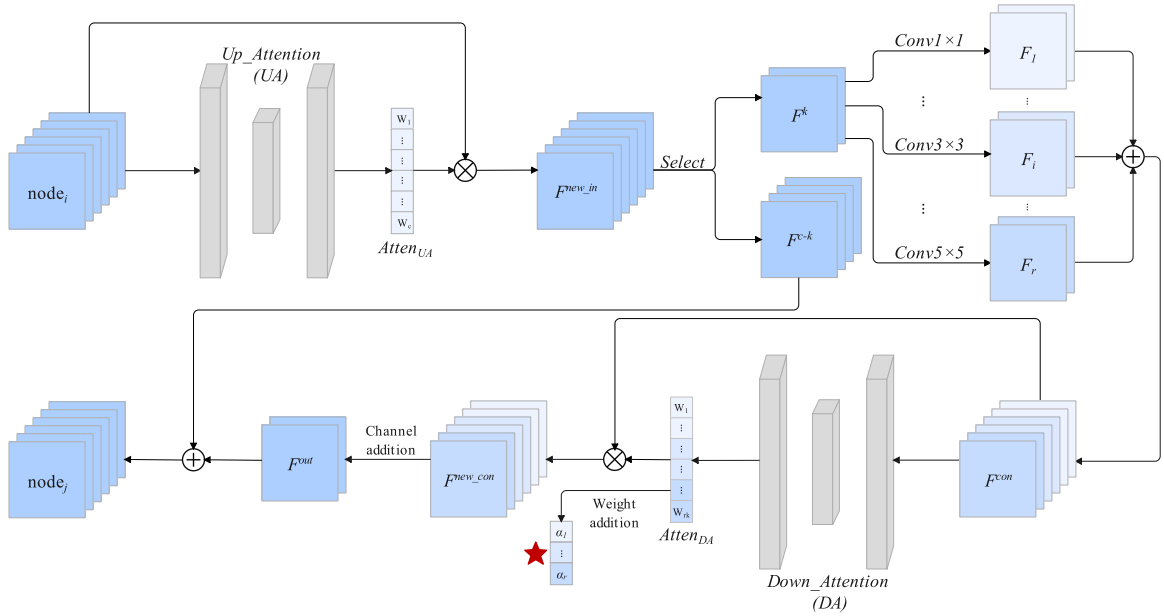


Fig. 2. Detailed functional block diagram of the operations occurring between two generic nodes of the DAMGAN supernet. node_i and node_j are two nodes in the supernet, such as 1 and 3 in Fig. 1. UA and DA are attention modules. F^{new_in} and F^{new_con} are attention feature maps, and F^k is the set of selected important feature maps. $F_{1\cdots r}$ are the feature maps obtained by different candidate operations. The NAS is formulated as the detection of the best candidate operations for each pair of nodes. The best candidate operations from the supernet as a whole identify the final subnet design.

1×1, 3×3 and 5×5 and deeply separable convolution with kernel sizes of 3×3, 5×5 and 7×7.

The choice of these nine potential operations–three upsampling operations and six convolution operations–is informed by prior knowledge of GAN functioning and insights from previous NAS studies for GANs, such as [26], [27], [28]. Upsampling tasks provide complementary advantages in the generation process. 'Nearest neighbour sampling' preserves fidelity to the original data, 'bilinear interpolation sampling' ensures smoothness, suitable for capturing contours and morphology, and 'deconvolution sampling' uses a learning-based approach for intricate detail generation. The convolution tasks include standard and depthwise separable convolutions, chosen for their ability to enhance feature extraction with reduced computational complexity. Different sizes of convolution kernels capture distinct features during the extraction process.

The candidate operations are applied to the feature maps selected by UA, producing corresponding output feature maps. These outputs are then processed by DA to weigh the importance of each candidate operation. The function of DA and UA will be explained in detail in Sections II-B and II-C, respectively.

Fig. 2 provides a detailed scheme of all operations occurring between two generic nodes, node_i and node_j. This scheme summarises the novelty of DAMGAN. Without a loss of generality, we show in Fig. 2 the case of a pair of nodes that excludes node 0, i.e., two nodes interconnected by convolution operations. For the sake of brevity, we show only three convolutions.

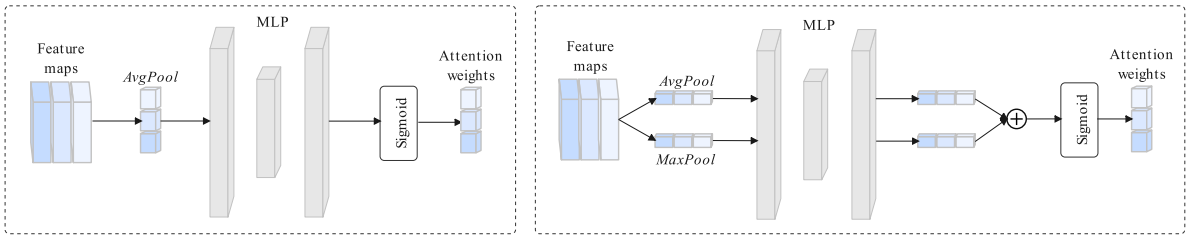
The departure node, node_i, contains a number of feature maps. These feature maps undergo UA, which generates a vector of weights whose elements represent the importance of each feature map. The weights are multiplied by the corresponding feature maps to generate a set of weighted feature maps F^{new_in} (see Section II-C for details). From the feature maps in F^{new_in}, the most important (comprising the set F^k) are selected to undergo candidate operations. The candidate operations are applied in parallel to the feature maps in F^k, generating a new set of output feature maps, F^{con}. The F^{con} set undergoes DA to assign the importance weights ($Atten_{DA}$) to the candidate operations. These weights are multiplied by the corresponding feature maps in F^{con} to generate the set F^{new_con}. At the same time, the weights in $Atten_{DA}$ are aggregated to detect the candidate operation that yields the best performance of the network (see Section II-B). Add the feature maps from the different operations in F^{new_con} and put them into the set F^{out}. The arrival node, node_j, contains the feature maps in F^{out} combined with those feature maps discarded by UA, i.e., those weighted (by UA weights) feature maps that did not undergo candidate operations.

B. Selection of Candidate Operations

To extract a subnet from the supernet, it is vital to select one candidate operation for each pair of nodes in a cell. As a preliminary operation, the generator supernet is trained by the Adam method [40], using the following loss functions:

$$\begin{aligned}\min_{\omega_G} V(G,D) &= \arg\min_{\omega_G} E_{z\sim p_z}\log(1-D(G(z))),\\ \max_{\omega_D} V(G,D) &= \arg\max_{\omega_D} E_{x\sim p_{data}}\log D(x)\\ &\quad + E_{z\sim p_z}\log(1-D\left(G(z)\right)),\end{aligned} \tag{1}$$

where ω_G is the weight of the generator supernet and ω_D is the weight of the discriminator.



(a) Attention mechanism DA (b) Attention mechanism UA

Fig. 3. Attention mechanisms DA and UA: (a) corresponds to DA in Fig. 2, (b) corresponds to UA in Fig. 2.

Once the supernet is trained, the optimal candidate operations are detected. Let r be the number of candidate operations between two adjacent nodes with connected edges, i.e., the departure and arrival nodes. When r candidate operations are applied to the feature maps of the departure node, r groups of feature maps $F_n(n = 1, 2, \ldots, r)$ are generated. Each output feature mapping is $F_n \in \mathbb{R}^{b\times k\times h\times w}$, where b is the batch size, k is the number of feature maps selected by the attention mechanism UA and $h \times w$ is the size of the feature map.

All the output feature maps produced by the application of all r candidate operations are connected together to generate the set of intermediate connected feature maps $F^{con} \in \mathbb{R}^{b\times(r\times k)\times h\times w}$. The attention module DA is then applied to F^{con} to learn the importance of each feature map. Thus, DA produces $r \times k$ attention weights (one for each feature map). The structure of DA is shown in Fig. 3(a). First, global average pooling is applied to F^{con} to compress the feature maps and obtain $F^{con}_{gap} \in \mathbb{R}^{b\times(r\times k)\times 1\times 1}$:

$$F^{con}_{gap} = \frac{1}{h \times w}\sum_{p=1}^{h}\sum_{q=1}^{w} F^{con}(p, q). \tag{2}$$

The set of compressed feature maps F^{con}_{gap} passes through a multi-layer perceptron network (MLP) composed of two fully connected layers, then through a sigmoid layer to obtain the attention weights $Atten_{DA}$. We use two fully connected layers with a bottleneck structure to reduce model complexity and improve generality. The first fully connected layer causes dimensionality reduction while the second fully connected layer is responsible for restoring the original dimension:

$$Atten_{DA} = \sigma\left(F_2\left(Relu\left(F_1\left(avgpool(F^{con}_{gap})\right)\right)\right)\right), \tag{3}$$

where F_1 denotes the first fully connected layer, F_2 denotes the second fully connected layer, σ is the sigmoid function; $Atten_{DA} \in \mathbb{R}^{b\times(r\times k)\times 1\times 1}$. If we consider, for example, the departure node 1 and the arrival node 3, for each batch size we have a vector of attention weights $Atten_{DA}$ of length 384 (6×64) elements, as there are $r = 6$ candidate operations and $k = 64$ feature maps selected by UA. Each group of k elements of $Atten_{DA}$ is then summed to calculate a number α_j, which signifies the importance of that set of output features:

$$\alpha_j = \sum_{i=(j-1)k+1}^{jk} Atten_{DA}(i), \tag{4}$$

Algorithm 1: Selection of Candidate Operation.

Input: Generator supernet G with attention module DA; training data D_{train} and discriminator D of AutoGAN
Output: Attention weights α^*
while not converged **do**
Compute the gradient of (1) by Adam and update the network weights ω_G and ω_D on the training data D_{train}
end while
Compute attention weights α by (4)
Compute maximum attention weight α^* by (6)
return α^*

where $j = 1, \ldots, r$. When the generator supernet has reached convergence, we can obtain the importance index of all candidate operations in the supernet (see the vector marked by the red star in Fig. 2).

To update the parameters of DA using a gradient method, i.e., Adam [40], we need to multiply the feature maps in F^{con} by their respective attention weights in $Atten_{DA}$. The result is the set of weighted feature maps $F^{new\_con} \in \mathbb{R}^{b\times(r\times k)\times h\times w}$:

$$F^{new\_con} = F^{con} \odot Atten_{DA}, \tag{5}$$

where $\odot$ indicates the multiplication of the elements $Atten_{DA}$ by the corresponding feature map.

Finally, the maximum attention weight α^* is selected from these attention weights, the candidate operation corresponding to α^* is identified as the most important candidate operation, and this operation is taken as the operation via which to construct the optimal subnet:

$$\alpha^* = \arg\max\left(\alpha_1, \alpha_2, \ldots, \alpha_r\right). \tag{6}$$

The specific algorithm is shown in Algorithm 1.

C. Selection of Feature Maps

Because the attention mechanism DA is added between each pair of adjacent nodes with connected edges in the generator supernet, the number of parameters to train in this supernet increases dramatically, with the potential for computationally costly supernet training. To mitigate this effect, we propose the inclusion of an up-attention mechanism UA between each pair of nodes, whose role is to select the most important features before multiple candidate operations are applied. Specifically, UA is

TABLE I
RESULTS OF DIFFERENT METHODS ON CIFAR-10 AND STL-10

Method	GPU days	Search space	Search method	CIFAR-10		STL-10	
				IS↑	FID↓	IS↑	FID↓
DCGAN [46]	-	-	Manual	6.64±0.14	37.70	-	-
WGAN-GP [7]	-	-	Manual	7.86±0.07	29.30	-	-
SNGAN [3]	-	-	Manual	8.22±0.05	21.70	9.16±0.12	40.10
GNGAN [4]	-	-	Manual	8.72±0.11	9.55	9.74±0.15	23.62
Progressive GAN [47]	-	-	Manual	8.80±0.05	18.33	-	-
Improving MMD GAN [48]	-	-	Manual	8.29	16.21	9.34	37.63
ProbGAN [49]	-	-	Manual	7.75	24.60	8.87±0.09	46.74
AGAN [24]	1200	20000	RL	8.29±0.09	30.50	9.23±0.08	52.70
AutoGAN [25]	2	$\sim 10^5$	RL	8.55±0.10	12.42	9.16±0.12	31.01
E^2GAN [50]	0.3	$\sim 10^5$	RL	8.51±0.13	11.26	9.51±0.09	25.35
EAGAN [26]	1.2	$\sim 10^{38}$	EA	8.81±0.10	9.91	10.44±0.08	22.18
DEGAS [51]	1.167	$\sim 10^8$	Gradient	8.37±0.08	12.01	9.71±0.11	28.76
AdversarialNAS [27]	1	$\sim 10^{38}$	Gradient	8.74±0.07	10.87	9.63±0.19	26.98
AlphaGAN [28]	0.13	$\sim 10^{11}$	Gradient	8.98±0.09	10.35	10.12±0.13	22.43
DAMGAN (our method)	**0.09**	$\sim 10^{11}$	Gradient	**8.99±0.08**	10.27	10.35±0.14	**22.18**

Bold values represent the best values for the column.

trained to select the top k most important feature maps from the departure node and apply the candidate operations to them. Not only does UA enhance the training speed of the supernet, as it reduces the number of features considered, but it also improves the stability of the training, as the supernet is better able to learn the most important features of the images.

As shown in Fig. 2, a feature maps selection process is performed to obtain F^{in} (where $F^{in} \in \mathbb{R}^{b\times c\times h\times w}$ and c is the number of feature maps in the departure node, node_i) before the candidate operations are applied to the feature maps in node_i (where $F^{select} \in \mathbb{R}^{b\times k\times h\times w}$). These operations are shown in Fig. 3(b). Global average pooling and global maximum pooling are first applied to the feature maps F^{in} of node_i, resulting in two sets of compressed feature maps; these are passed through an MLP consisting of two fully connected layers, then through a sigmoid layer to obtain the attention weights $Atten_{UA}$:

$$Atten_{UA} = \sigma(F_2(Relu(F_1(avgpool(F^{in}))))) + F_2(Relu(F_1(maxpool(F^{in}))))), \quad (7)$$

where $avgpool$ is the global average pooling, $maxpool$ is the global maximum pooling, F_1 denotes the first fully connected layer, F_2 denotes the second fully connected layer and σ is the sigmoid function; $Atten_{UA} \in \mathbb{R}^{b\times c\times 1\times 1}$. To update UA parameters by backpropagation, i.e., Adam [40], the feature maps in F^{in} must be multiplied by their respective attention weights in $Atten_{UA}$. The result is the set of weighted feature maps $F^{new\_in} \in \mathbb{R}^{b\times\ c\times h\times w}$:

$$F^{new\_in} = F^{in} \odot Atten_{UA}, \quad (8)$$

where $\odot$ indicates the multiplication of the elements $Atten_{UA}$ by the corresponding feature map. Finally, the top k feature maps F^k from F^{new_in} are selected according to the size of the attention weights $Atten_{UA}$, and the candidate operations are applied to F^k:

$$F^k = Select\left(F^{new\_in}, Atten_{UA}, k\right), \quad (9)$$

where $Select$ is the selection function whose detailed pseudo-code is shown in Algorithm 2. The feature maps that are not selected are then combined with F^{out} as the feature maps in the arrival node, $node_j$.

Algorithm 2: Selection of Feature Maps.

Input: feature maps set F^{new_in}, attention weight $Atten_{UA}$, top k feature maps
Output: Selected feature maps F^k

```
s ← 0
F_max ← top k weightin Atten_UA
while s < c do
  if Atten_s ∈ F_max then
    M_s^(i,j) ← 1 {Where M_s is used to record the
     position of the feature map to be selected in
     Atten_UA. If the feature map is to be selected, the
     value of M_s is 1; otherwise the value of M_s is 0.}
  else
    M_s^(i,j) ← 0
  end if
  s ← s + 1
end while
s, n ← 0
while s < c do
  if M_s^(i,j) then
    F_n^k ← F_(i,j)^new_in
    n ← n + 1
  end if
end while
return F^k
```

Algorithm 3: Search Phase.

Input: training iterations N; discriminator update steps per iteration n_D; batch size of generator supernet M_G; batch size of discriminator M_D; Adam optimiser parameters $\alpha_{lr}, \beta_1, \beta_2$; initial discriminator and generator parameters w_D, w_G
Output: Optimal subnet
for $i = 1 \rightarrow N$ **do**
  for $k = 1 \rightarrow n_D$ **do**
    Sample a batch of noise z and real data x
    $g_{w_D} \leftarrow \nabla_{w_D} \left[\frac{1}{m}\sum_{i=1}^{m} \log D(x^{(i)}) + \frac{1}{m}\sum_{i=1}^{m} \log(1 - D(G(z^{(i)})))\right]$
    Update w_D with Adam
  end for
  Sample a batch of noise z
  $g_{w_G} \leftarrow \nabla_{w_G} \frac{1}{m}\sum_{i=1}^{m} \log D(G(z^{(i)}))$
  Update w_G with Adam
  end for
The corresponding operation is selected according to the α^* obtained by Algorithm 1 and the optimal subnet is constructed.
  return optimal subnet

D. Selection of the Optimal Subnet

The supernet of the described architecture is trained through a standard gradient-based approach. First we need to construct the generator supernet and the discriminator network, in order to make a fair comparison with other methods [25], [26], [27], [28], we construct the same discriminator network architecture as the discriminator architecture in AutoGAN. The training process of GANs is actually a process of maximisation and minimisation optimisation of the same objective function by both the generator and the discriminator, i.e., the generator wants to minimise the objective function, the discriminator wants to maximise the objective function. Therefore, we need to optimise the generator and the discriminator alternately. Specifically, we compute the gradient of the generator and the discriminator based on their objective values, and then optimise their network weights alternately according to a predefined optimiser. When the supernet training converges, the optimal subnet is immediately determined when, for each pair of interconnected nodes, the best candidate operation is selected according to (6). The pseudocode detailing the logic of the subnet extraction is shown in Algorithm 3. It is noteworthy that, whereas AdversarialNAS and AlphaGAN are based on a two-layer optimisation strategy, thanks to the proposed architecture, DAMGAN requires only one optimisation level, i.e., the training of the supernet. This is achieved using DA. Furthermore, UA serves the important purpose of controlling the computational cost of the optimisation process. The use of two concurrent attention mechanisms in the optimisation is an important novelty of the proposed approach.

III. Experimental Results

This section presents the experimental setup and results of this study, as follows. First, Section III-A introduces the datasets and performance metrics. Section III-B then describes the implementation details and parameters of DAMGAN. Sections III-C and III-D summarise the comparison results. Sections III-E shows experimental results on a large size dataset. Finally, Section III-F details the ablation study.

A. Datasets and Performance Metrics

To guarantee a fair comparison, we have reproduced the experimental setup of NAS for GAN used in [24], [25], [26], [27], [28]. Specifically, we validated our approach on the CIFAR-10 [41] and STL-10 [42] datasets. CIFAR-10 contains a total of 60,000 images of spatial resolution 32×32, in ten different categories. STL-10 contains 100, 500 images of resolution 96×96, which we resized to 48×48 to ensure fair comparison with other methods.

To facilitate comparison with other methods, we evaluated the quality of our generated images using IS [43] and FID [44], calculated via 50,000 images randomly generated from the model.

B. Implementation Details

As mentioned above, we employed the Adam optimiser [40] to perform the gradient descent. The loss function of the GANs is the hinge loss [45]. In our implementation, the search phase used the same discriminator as used in AutoGAN [25]. The hyperparameters of the optimisers used to train the weights of the generator and discriminator networks were set to $\beta_1 = 0$, $\beta_2 = 0.9$; the learning rate of the generator was set to 0.0002; and the learning rate of the discriminator was set to 0.0002. The batch size was set to 64 for both the generator and discriminator, with the number of feature maps set to 256 for the generator and to 128 for the discriminator. In the retraining phase, the discriminator used was the same as in the search phase. The learning rate for both the generator and discriminator was set to 0.0002, the batch size was set to 128 for the generator and 64 for the discriminator, and the number of feature maps was set to 256 for the generator and 128 for the discriminator. Our addition of a second attention mechanism to the supernet pays more attention to the most important portion of the input features, which means that the training of the supernet can converge quickly. We therefore trained the supernet for approximately 100 epochs in the search phase and 300 epochs in the retraining phase. The experiments were performed on an NVIDIA 3090 GPU.

C. Results on CIFAR-10

The subnet designed by DAMGAN for CIFAR-10 is shown in Fig. 4.

When supernet is trained by CIFAR-10. Among the upsampling operations, transposed convolution ('deconvolution sampling') appears to be the preferred operation. This is because we adopt the second attention mechanism in the generator supernet and it pays more attention to the most important feature maps, and transposed convolution can extract important features. The operation of the second preference is 'bilinear interpolation

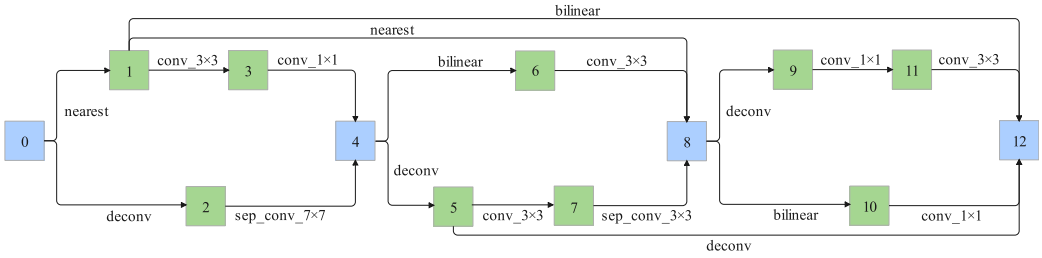


Fig. 4. Generator structure designed by DAMGAN on the CIFAR-10 dataset.

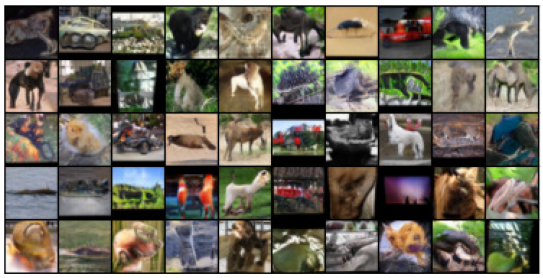


Fig. 5. Randomly generated images on CIFAR-10.

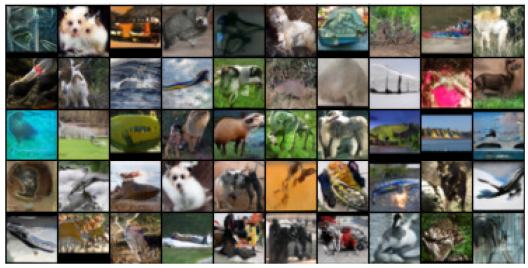


Fig. 6. Randomly generated images on STL-10.

Fig. 7. Randomly generated images on CelebA.

sampling', and the last is 'nearest neighbour sampling'. Regarding convolution operations, DAMGAN appears to be prone to select convolutions with smaller convolution kernels. A possible reason is that the feature maps in the supernet are relatively small and effective for feature extractions such as '3×3 convolution' and '1×1 convolution'. Thus, the algorithm appears to adapt to these convolution sizes. Another feature of our generator network is that transposed convolution is not used on two consecutive layers, probably because successive transposed convolutions tend to lead to information loss, which is fatal to the generation task.

Table I displays the results of DAMGAN on the CIFAR-10 dataset. In Table I, we compare DAMGAN with 13 other state-of-the-art algorithms. Each of them was run with its original parameter settings, as reported in the corresponding source. Under the same regularisation conditions, the network architecture designed by DAMGAN obtains relatively good results when compared with the 13 competitors. It can be seen that, on CIFAR-10, DAMGAN achieves a competitive FID (10.27) and the best performance in terms of IS (8.99 ± 0.08). It is worth remarking that this IS is the highest ever achieved by a NAS for GANs on CIFAR-10. Furthermore, our method is the computationally cheapest method by far among the NAS algorithms considered in this study, i.e., DAMGAN requires only 0.09 GPU days (just over two hours on a single GPU).

Fig. 5 displays 50 randomly generated images. It can be seen that DAMGAN can generate a variety of images that appear realistic.

D. Portability of Architecture

In order to verify the portability of the network architectures searched by DAMGAN, we directly migrated the generator architecture searched by DAMGAN on the CIFAR-10 dataset to the STL-10 dataset for training. Then, we used a total of 105,000 unlabelled images to train this network, training the generator for 240 epochs. The batch sizes used to optimise the generator and discriminator were set to 128 and 64, respectively. The number of feature maps was set to 256 for the generator and 128 for the discriminator. The learning rate was set to 0.0002 for both the generator and discriminator.

The quantitative results are shown in Table I too. It can be seen that DAMGAN achieves a relatively good IS (10.35 ± 0.14) and the best FID (22.18) on STL-10. Given that DAMGAN requires the smallest design time, we may conclude that DAMGAN is very promising even on large and complex datasets. To demonstrate DAMGAN's ability to generate diverse and realistic images, we present qualitative results in Fig. 6.

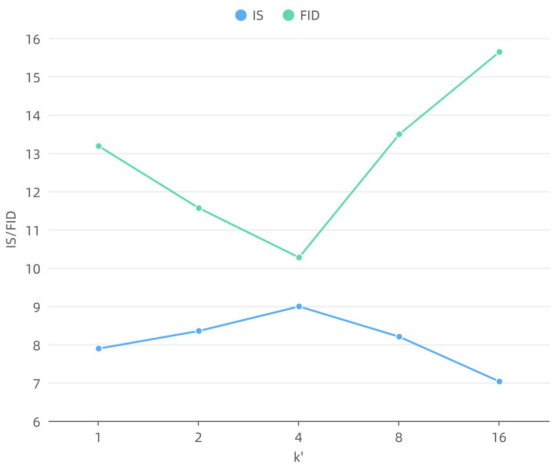


Fig. 8. Effect of k' value on IS and FID.

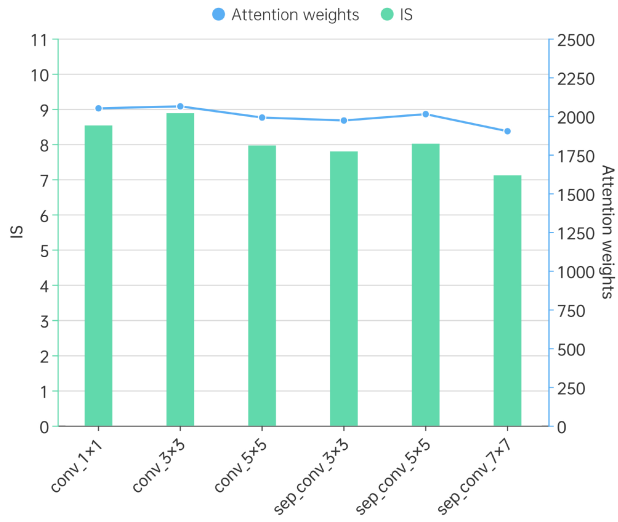


Fig. 9. IS values corresponding to different attention weights.

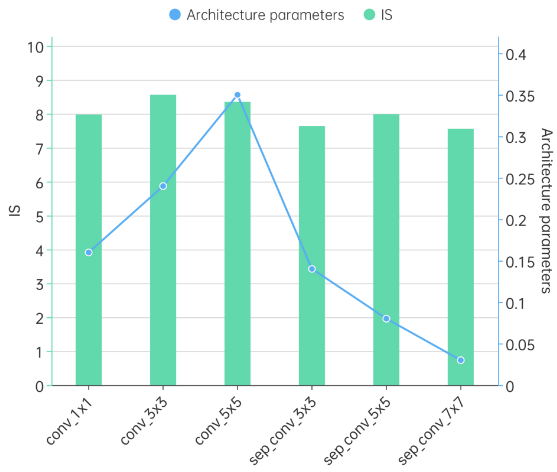


Fig. 10. IS values corresponding to different architectural parameters.

E. Results on CelebA

To validate that the architectures searched through our method can generate larger-sized images, we trained the architecture found on the CIFAR-10 dataset using the CelebA dataset. Specifically, we utilized a total of 202,599 face images for training the network architecture. The batch size was set to 32 for both the generator and the discriminator, and the learning rate was set to 0.0002 for both. The generator had 256 feature maps, while the discriminator had 128. In Fig. 7, the generated face images are displayed upon convergence of the network training, and they are of size 64×64. The figure illustrates that the network architecture searched on the CIFAR-10 dataset using our method can be effectively applied to the CelebA dataset, generating higher-resolution images.

F. Ablation Studies

Ablation experiments were performed to verify the efficiency of the proposed techniques.

1) Analysis of the Feature Maps Selection Ratio $\frac{1}{k'}$: We verified the influence of feature maps selection ratio k' on the performance of the optimal subnet. We experimented with different values of k': specifically, we set $k' = \{1, 2, 4, 8, 16\}$ to search for optimal subnets, then retrained those optimal subnets, before finally comparing the IS and FID values of the images they generated. Fig. 8 shows the best IS and FID values when different values of k' are employed. Initially, as k' increases, the IS value increases and the FID value decreases. The image generated by the searched network architecture achieves the best quality when $k' = 4$. This indicates that our feature maps selection mechanism can search for network architectures with good performance most efficiently at this value of k'. However, as k' continues to increase, the IS and FID values worsen. This indicates that the number of feature maps applied to the candidate operation cannot be too small; if this occurs, the network is unable to obtain sufficient feature information during training, and thus cannot achieve a good network architecture.

2) Analysis of Attention Weights α^*: We investigated the relative suitability of attention weights (as in the proposed DAMGAN) and architectural parameters (as in DARTS and modern competitors based on DARTS) as indicators of the importance of candidate operations, conducting experiments on the CIFAR-10 dataset. We performed two searches, one guided by attention weights and the other guided by architectural parameters, and, at the end of each search, constructed the optimal subnet as identified by the given method. In the example edge $1 \rightarrow 3$ in the first cell, we set each of the six candidate operations in the original $1 \rightarrow 3$ as the actual connection edge $1 \rightarrow 3$ in turn, and trained the subnet. The attention weights or architectural parameters of the edges actually used and the IS values obtained from their corresponding training were recorded.

Fig. 9 shows the different IS values obtained from the subnet training constructed according to different attention weights. '$Conv\_3 \times 3$' has the largest attention weight, which corresponds to the highest IS score. Fig. 10 shows the different IS values obtained from the subnet training constructed according to different architectural parameters. '$Conv\_5 \times 5$' has the largest value of the architectural parameter, but the corresponding IS score here is not the highest. This demonstrates that

attention weight is more suitable than architectural parameter as an indicator of the importance of candidate operations and as an estimator of the performance of the final architecture. Overall, Figs. 9 and 10 appear to indicate that attention weights are much more correlated to IS than architectural parameters.

IV. Conclusion and Future Work

This paper proposes a novel NAS method for designing GAN architectures, i.e., a model for image generation. The proposed method, DAMGAN, makes use of a supernet and extracts subnets as the candidate generator architectures. Different from established NAS methods for GANs, DAMGAN replaces the architectural parameters used in previous studies with attention weights determined by a preliminary attention mechanisms. The experiments clearly demonstrate that attention weights indicate the relative importance of operations far better than do architectural parameters. To improve the stability and search speed of supernet training, we propose a strategy to select the feature maps for candidate operations that are based on a second attention mechanism. The second mechanism enables control of the computational cost of the supernet training.

Experimental results show that our method achieves excellent performance in the field of NAS for GANs and completes the search of architectures in only 0.09 GPU days on the CIFAR-10 dataset. On this dataset, DAMGAN also achieved a new IS record for NAS methods. Furthermore, our experiments on the STL-10 dataset indicate the portability potential of this NAS logic: the generator designed on the relatively small CIFAR-10 dataset performed extremely well after retraining on the larger STL-10 dataset.

To enhance the efficiency of GANs architecture search, in the future we plan to investigate and refine the search space. Our strategy involves implementing a supernet pruning mechanism, eliminating suboptimal operations beforehand for a quicker search. Additionally, we aim to improve the detail generation capability of our network architectures. These efforts will contribute to the swift discovery of high-performance GANs architectures, fostering their broader applicability across domains.

References

[1] I. Goodfellow et al., "Generative adversarial networks," *Commun. ACM*, vol. 63, no. 11, pp. 139–144, 2020.

[2] J. He et al., "Finger vein image deblurring using neighbors-based binary-GAN (NB-GAN)," *IEEE Trans. Emerg. Topics Computat. Intell.*, vol. 7, no. 2, pp. 295–307, Apr. 2023.

[3] L. Shen, J. Yan, X. Sun, B. Li, and Z. Pan, "Wavelet-based self-attention GAN with collaborative feature fusion for image inpainting," *IEEE Trans. Emerg. Topics Comput. Intell.*, vol. 7, no. 6, pp. 1651–1664, Dec. 2023.

[4] G. Zhong, W. Ding, L. Chen, Y. Wang, and Y. Yu, "Multi-scale attention generative adversarial network for medical image enhancement," *IEEE Trans. Emerg. Topics Comput. Intell.*, vol. 7, no. 4, pp. 1113–1125, Aug. 2023.

[5] X. Mao, Q. Li, H. Xie, R. Y. Lau, Z. Wang, and S. Paul Smolley, "Least squares generative adversarial networks," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2017, pp. 2794–2802.

[6] M. Arjovsky, S. Chintala, and L. Bottou, "Wasserstein generative adversarial networks," in *Proc. 34th Int. Conf. Mach. Learn.*, 2017, pp. 214–223.

[7] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. C. Courville, "Improved training of wasserstein GANs," in *Proc. 31st Int. Conf. Neural Inf. Process. Syst.*, 2017, pp. 5769–5779.

[8] Z. Li, P. Xia, R. Tao, H. Niu, and B. Li, "A new perspective on stabilizing GANs training: Direct adversarial training," *IEEE Trans. Emerg. Topics Comput. Intell.*, vol. 7, no. 1, pp. 178–189, Feb. 2023.

[9] T. Karras, S. Laine, and T. Aila, "A style-based generator architecture for generative adversarial networks," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 4401–4410.

[10] T. Karras, S. Laine, M. Aittala, J. Hellsten, J. Lehtinen, and T. Aila, "Analyzing and improving the image quality of StyleGAN," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 8110–8119.

[11] Y. Chen, T. Yang, X. Zhang, G. Meng, X. Xiao, and J. Sun, "DETNAS: Backbone search for object detection," in *Proc. 33rd Int. Conf. Neural Inf. Process. Syst.*, 2019, pp. 6642–6652.

[12] G. Ghiasi, T. Lin, and Q. V. Le, "NAS-FPN: Learning scalable feature pyramid architecture for object detection," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 7036–7045.

[13] J. Peng, M. Sun, Z. Zhang, T. Tan, and J. Yan, "Efficient neural architecture transformation search in channel-level for object detection," *Proc. 33rd Int. Conf. Neural Inf. Process. Syst.*, 2019, pp. 14335–14344.

[14] Y. Bi, B. Xue, P. Mesejo, S. Cagnoni, and M. Zhang, "A survey on evolutionary computation for computer vision and image analysis: Past, present, and future trends," *IEEE Trans. Evol. Comput.*, vol. 27, no. 1, pp. 5–25, Feb. 2023.

[15] Y. Sun, B. Xue, M. Zhang, and G. G. Yen, "Evolving deep convolutional neural networks for image classification," *IEEE Trans. Evol. Comput.*, vol. 24, no. 2, pp. 394–407, Apr. 2020.

[16] Y. Sun, B. Xue, M. Zhang, G. G. Yen, and J. Lv, "Automatically designing CNN architectures using the genetic algorithm for image classification," *IEEE Trans. Cybern.*, vol. 50, no. 9, pp. 3840–3854, Sep. 2020.

[17] Y. Xue, Y. Wang, J. Liang, and A. Slowik, "A self-adaptive mutation neural architecture search algorithm based on blocks," *IEEE Comput. Intell. Mag.*, vol. 16, no. 3, pp. 67–78, Aug. 2021.

[18] Y. Xue, C. Chen, and A. Słowik, "Neural architecture search based on a multi-objective evolutionary algorithm with probability stack," *IEEE Trans. Evol. Comput.*, vol. 27, no. 4, pp. 778–786, Aug. 2023.

[19] Y. Xue, C. Lu, F. Neri, and J. Qin, "Improved differentiable architecture search with multi-stage progressive partial channel connections," *IEEE Trans. Emerg. Topics Comput. Intell.*, vol. 8, no. 1, pp. 32–43, Feb. 2024.

[20] C. Liu et al., "Auto-DeepLab: Hierarchical neural architecture search for semantic image segmentation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 82–92.

[21] Z. Lu, R. Cheng, S. Huang, H. Zhang, C. Qiu, and F. Yang, "Surrogate-assisted multiobjective neural architecture search for real-time semantic segmentation," *IEEE Trans. Artif. Intell.*, vol. 4, no. 6, pp. 1602–1615, Dec. 2023.

[22] L. Chen et al., "Searching for efficient multi-scale architectures for dense image prediction," in *Proc. 32nd Int. Conf. Neural Inf. Process. Syst.*, 2018, pp. 8713–8724.

[23] Y. Zhang, Z. Qiu, J. Liu, T. Yao, D. Liu, and T. Mei, "Customizable architecture search for semantic segmentation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 11641–11650.

[24] H. Wang and J. Huan, "AGAN: Towards automated design of generative adversarial networks," 2019, *arXiv:1906.11080*.

[25] X. Gong, S. Chang, Y. Jiang, and Z. Wang, "AutoGAN: Neural architecture search for generative adversarial networks," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2019, pp. 3224–3234.

[26] G. Ying, X. He, B. Gao, B. Han, and X. Chu, "EAGAN: Efficient two-stage evolutionary architecture search for GANs," in *Proc. Eur. Conf. Comput. Vis.*, 2022, pp. 37–53.

[27] C. Gao, Y. Chen, S. Liu, Z. Tan, and S. Yan, "AdversarialNAS: Adversarial neural architecture search for GANs," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 5680–5689.

[28] Y. Tian, L. Shen, G. Su, Z. Li, and W. Liu, "AlphaGAN: Fully differentiable architecture search for generative adversarial networks," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 44, no. 10, pp. 6752–6766, Oct. 2022.

[29] Q. Lin, Z. Fang, Y. Chen, K. C. Tan, and Y. Li, "Evolutionary architectural search for generative adversarial networks," *IEEE Trans. Emerg. Topics Comput. Intell.*, vol. 6, no. 4, pp. 783–794, Aug. 2022.

[30] H. Pham, M. Guan, B. Zoph, Q. Le, and J. Dean, "Efficient neural architecture search via parameters sharing," in *Proc. 35th Int. Conf. Mach. Learn.*, 2018, pp. 4095–4104.

[31] Y. Sun, X. Sun, Y. Fang, G. G. Yen, and Y. Liu, "A novel training protocol for performance predictors of evolutionary neural architecture search algorithms," *IEEE Trans. Evol. Comput.*, vol. 25, no. 3, pp. 524–536, Jun. 2021.

[32] Y. Liu, Y. Tang, and Y. Sun, "Homogeneous architecture augmentation for neural predictor," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2021, pp. 12249–12258.
[33] H. Liu, K. Simonyan, and Y. Yang, "DARTS: Differentiable architecture search," in *Int. Conf. Learn. Representations*, 2019.
[34] Y. Xue and J. Qin, "Partial connection based on channel attention for differentiable neural architecture search," *IEEE Trans. Ind. Informat.*, vol. 19, no. 5, pp. 6804–6813, May 2023.
[35] X. Dong and Y. Yang, "Searching for a robust neural architecture in four gpu hours," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 1761–1770.
[36] G. Li, G. Qian, I. C. Delgadillo, M. Muller, A. Thabet, and B. Ghanem, "SGAS: Sequential greedy architecture search," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 1620–1630.
[37] X. Chu, T. Zhou, B. Zhang, and J. Li, "Fair DARTS: Eliminating unfair advantages in differentiable architecture search," in *Proc. Eur. Conf. Comput. Vis.*, 2020, pp. 465–480.
[38] Z. Sun et al., "AGNAS: Attention-guided micro and macro-architecture search," in *Proc. 39th Int. Conf. Mach. Learn.*, 2022, pp. 20777–20789.
[39] F. Briggs, G. R. Mangun, and W. M. Usrey, "Attention enhances synaptic efficacy and the signal-to-noise ratio in neural circuits," *Nature*, vol. 499, no. 7459, pp. 476–480, 2013.
[40] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proc. Int. Conf. Learn. Representations*, 2015, pp. 1–13.
[41] A. Krizhevsky, "Learning multiple layers of features from tiny images," M.S. thesis, Univ. Tront, 2009.
[42] A. Coates, A. Ng, and H. Lee, "An analysis of single-layer networks in unsupervised feature learning," in *Proc. 14th Int. Conf. Artif. Intell. Stat.*, 2011, pp. 215–223.
[43] T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen, "Improved techniques for training GANs," in *Proc. 30th Int. Conf. Neural Inf. Process. Syst.*, 2016, pp. 2234–2242.
[44] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter, "GANs trained by a two time-scale update rule converge to a local nash equilibrium," in *Proc. 31st Int. Conf. Neural Inf. Process. Syst.*, 2017, pp. 6629–6640.
[45] J. H. Lim and J. C. Ye, "Geometric GAN," 2017, *arXiv:1705.02894*.
[46] A. Radford, L. Metz, and S. Chintala, "Unsupervised representation learning with deep convolutional generative adversarial networks," in *Proc. Int. Conf. Learn. Representations*, 2016, pp. 1–16.
[47] T. Karras, T. Aila, S. Laine, and J. Lehtinen, "Progressive growing of GANs for improved quality, stability, and variation," in *Proc. Int. Conf. Learn. Representations*, 2018, pp. 1–26. [Online]. Available: https://openreview.net/forum?id=Hk99zCeAb
[48] W. Wang, Y. Sun, and S. Halgamuge, "Improving MMD-GAN training with repulsive loss function," in *Proc. Int. Conf. Learn. Representations*, 2018, pp. 1–24.
[49] H. He, H. Wang, G. Lee, and Y. Tian, "ProbGAN: Towards probabilistic GAN with theoretical guarantees," in *Proc. Int. Conf. Learn. Representations*, 2019, pp. 1–28.
[50] Y. Tian et al., "Off-policy reinforcement learning for efficient and effective gan architecture search," in *Proc. Eur. Conf. Comput. Vis.*, 2020, pp. 175–192.
[51] S. Doveh and R. Giryes, "Degas: Differentiable efficient generator search," *Neural Comput. Appl.*, vol. 33, pp. 17173–17184, 2021.

Yu Xue(Senior Member, IEEE) received the Ph.D. degree in application technology of computer from the School of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, Nanjing, China, in 2013. He is currently a Professor with the School of Software, Nanjing University of Information Science and Technology. From 2016 to 2017, he was a Visiting Scholar with the School of Engineering and Computer Science, Victoria University of Wellington, Wellington, New Zealand. From 2017 to 2018, he was a Research Scholar with the Department of Computer Science and Engineering, Michigan State University, East Lansing, MI, USA. His research interests include deep learning, evolutionarycomputation, machine learning, and computer vision.

Kun Chen is currently working toward the M.Sc. degree with the School of Software, Nanjing University of Information Science and Technology, Nanjing, China. His research interests include deep learning, image generation, evolutionary neural architecture search, and evolutionary generative adversarial networks.

Ferrante Neri (Senior Member, IEEE) received the Laurea and Ph.D. degrees in electrical engineering from the Technical University of Bari, Bari, Italy, in 2002 and 2007, respectively, and the the Ph.D. degree in scientific computing and optimization, and the D.Sc. degree in computational intelligence from the University of Jyväskylä, Jyväskylä, Finland, in 2007 and 2010, respectively. Between 2009 and 2014, he was an Academy Research Fellow with the Academy of Finland to lead the project "Algorithmic Design Issues in Memetic Computing." Between 2012 and 2019, he was with De Montfort University, Leicester, U.K., and with the University of Nottingham, Nottingham, U.K., between 2019 and 2022. Since 2022, he has been with the University of Surrey, Guildford, USA, as a Full Professor of machine learning and artificial intelligence and the Head with the Nature Inspired Computing and Engineering (NICE) Research Group. His research focuses on metaheuristic optimisation with applications in the context of machine learning.