Hindawi
Security and Communication Networks
Volume 2021, Article ID 6621760, 9 pages
https://doi.org/10.1155/2021/6621760

Research Article

Genetic Feature Fusion for Object Skeleton Detection

Yang Qiao, Yunjie Tian, Yue Liu, and Jianbin Jiao

School of Electronic, Electrical, and Communication Engineering, University of Chinese Academy of Sciences, Beijing 100049, China

Correspondence should be addressed to Jianbin Jiao; jiaojb@ucas.ac.cn

Received 31 December 2020; Revised 28 January 2021; Accepted 24 February 2021; Published 9 March 2021

Academic Editor: Liguo Zhang

Object skeleton detection requires the convolutional neural networks to recognize objects and their parts in the cluttered background, overcome the image definition degradation brought by the pooling layers, and predict the location of skeleton pixels in different scale granularity. Most existing object skeleton detection methods take great efforts into the designing of side-output networks for multiscale feature fusion. Despite the great progress achieved by them, there are still many problems that hinder the development of object skeleton detection, such as the manually designed network is labor-intensive and the network initialization depends on models pretrained on large-scale datasets. To alleviate these issues, we propose a genetic NAS method to automatically search on a newly designed architecture search space for adaptive multiscale feature fusion. Furthermore, we introduce a symmetric encoder-decoder search space based on reversing the VGG network, in which the decoder can reuse the ImageNet pretrained model of VGG. The searched networks improve the performance of the state-of-the-art methods on commonly used skeleton detection benchmarks, which proves the efficacy of our method.

1. Introduction

Skeleton is an inherent visual descriptor of natural objects, which contains rich shape semantics of the objects. As compact topological representations of objects, the object skeletons boost the research of hand gesture recognition [1], human pose estimation [2], text detection [3], etc.

In the deep learning era, object skeleton detection methods are typically implemented by convolutional neural networks (CNNs) in an end-to-end manner and usually formulated as pixel-wise binary classification tasks. The essential difficulty of object skeleton detection can be summarized as how to simultaneously detect object skeletons with various scales since specific convolutional layers have limited receptive field size and lack of expression ability.

Based on the consensus that features from shallow layers contain low-level fine details while features from deep layers have rich high-level semantics, existing object detection works take great efforts on designing side-output networks for multiscale feature fusion. To name a few, holistically nested edge detection (HED) [4] pioneeringly detected object skeletons using abstracted multiscale representation extracted by the CNN, which are fused in a hierarchical parallel structure. Fusing scale-associated deep side-outputs (FSDS) [5] utilized scale-associated ground truth to supervise the network in a divided-and-conquer fashion, where deep side-output features with specific scale were fused gradually to predict skeleton maps in each layer. Side-output residual network (SRN) [6, 7] adopted side-output residual units to integrate the feature between adjacent stages from deep to shallow. Hi-Fi [8] introduced a bilateral feature integration mechanism to fuse features of different scales more densely. Despite great progress made by existing works on multiscale feature fusion, one limitation remains that the existing architectures are still hand-designed. The decoder network has no pretrained parameters. These disadvantages hinder the development of object skeleton detection to a certain extent.

In this paper, inspired by the remarkable success of neural architecture search (NAS) in many computer vision fields, such as image identification, object detection, and semantic segmentation, we propose a novel NAS-based object skeleton detection method, termed as genetic feature

fusion (GFF). Unlike the architectures of previous approaches, GFF introduces NAS to search the network architecture for adaptive feature fusion automatically. The follow-ups of HED [4] use the networks which are commonly used in classification, such as VGG [9], but it is difficult for these networks to have good adaptability on object skeleton detection. Guided by this view, we propose to expand the side-output architectures of the existing network and search the expanded search space to adaptively get the most suitable network architecture for object skeleton detection. In order to make full use of the pretrained parameters of these existing networks, the decoder is also reversed from the backbone (Figure 1). We utilize a genetic algorithm to search the expanded search space and get some novel networks which achieve state-of-the-art performance.

The contributions of this work are summarized as follows:

(i) We propose a novel NAS-based object skeleton detection method referred to as genetic feature fusion (GFF), which can make full use of pretrained parameters of existing networks both on encoder and decoder networks and adaptively fuse the multiscale features extracted in different convolutional layers

(ii) We introduce a search space expanded from the existing networks, which is designed to bridge the gap between the classification networks and object skeleton detection networks

(iii) We improve the state-of-the-art performance and demonstrate the effectiveness of our GFF on commonly used skeleton detection benchmarks

2. Related Work

Object skeleton detection is of great significance in the field of computer vision that skeletons reveal the inherent topological attributes of objects. In this section, we first review feature fusion architectures and then the object skeleton detection methods and finally summarize NAS technology.

2.1. Object Skeleton Detection. Convolutional neural networks (CNNs) play an absolutely dominant role in image-to-mask tasks. However, CNN itself suffers inherent contradictions, that is, good results require fine-grained scales from shallow features and abstract semantics from deep features. For alleviating this, researchers introduced and extensively studied feature fusion, which combines multiscale features to produce results. Object skeleton detection (symmetry detection) is an important image-to-mask task in computer vision. It helps to understand advanced features of objects in images, such as the topological property.

Early methods usually utilize geometric modelling, e.g., morphological image operation, to detect the skeleton of objects. Image segmentation procedures have to be performed as preprocessing when dealing with colour images. These methods always have good interpretability but poor performance.

Entering the era of deep learning, the performance of object skeleton detection has been improving greatly after the proposal of HED [4], which is a pioneer work to formulate the skeleton detection problem as a pixel-wise binary classification task. HED developed a parallel multiscale feature fusion with the help of a convolution neural network. Most methods proposed in the follow-up also treated object skeleton detection as a pixel-wise binary classification task and improved HED from the perspective of feature fusion. As shown in Figure 2, many popular works focused on the feature fusion to improve the performance.

Given scale-associated ground truth, fusing scale-associated deep side-output (FSDS) [5] learned the skeleton of objects using multiple network stages, which correspond to different scales. SRN [6] proposed a side-output residual network to expand the feature space to fit the errors between the output feature and the skeleton ground truth. Side-output residual units build short connections and send the deep coarse-grained features to the shallow fine-grained layer. In this way, features of different scales are merged to generate better skeleton maps. RSRN [7] utilized dense side-output residual units to make full use of richer multiscale features. Like FSDS, Hi-Fi [8] also used scale-associated supervision and built a hierarchical feature integration mechanism between stages of the network for the better fusion of features. LSN [10] increased the independence of different layer features using the linear span network and linear span units under the linear span view. Liu et al. [11] designed an orthogonal decomposition network to enrich feature diversity. PSG [12] uses Gaussian mixture models to partition skeleton branches for parametric skeleton generalization.

2.2. Neural Architecture Search. The recent rise of automated machine learning (AutoML) has caused extensive research because it automatically optimizes hyperparameters, which saves expensive expert knowledge. As an important part of AutoML, neural architecture search (NAS) is committed to automating the process of network architecture design and has found better network structures than manually designed.

NAS has three core components: search strategy, search space, and evaluation strategy. Four search strategies are commonly used, including reinforcement learning, evolutionary algorithms, Bayesian optimization, and one-shot methods. Early NAS is mainly formulated by reinforcement- and evolutionary-based methods but endures huge computational costs. The search process of NASNet [13], based on reinforcement learning, took 1800 GPU days, while the evolutionary-based AmoebaNet [14] took 3150 GPU days. The reason for such a huge computational overhead is that the sampled network structure must be independently trained in the evaluation phase. In order to reduce the computational cost, the one-shot method is proposed, which adopts the strategy of weight sharing to train the supernet, and the sampled networks inherit the weight of the supernet to avoid pretraining during evaluation. As a special family of weight sharing, gradient-based methods represented by DARTS [15] continue the discrete search space and achieve the purpose of searching

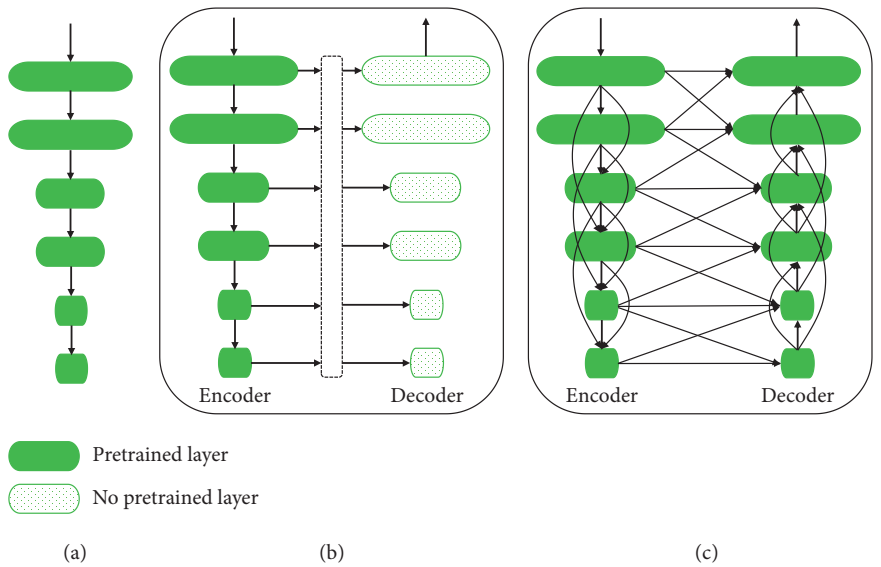

Figure 1: (a) A pretrained network migrated from classification; (b) the existing methods, of which the encoder directly utilized (a), and the decoder, after feature integration, used random initialized parameters; (c) our GFF with the expanded search space.

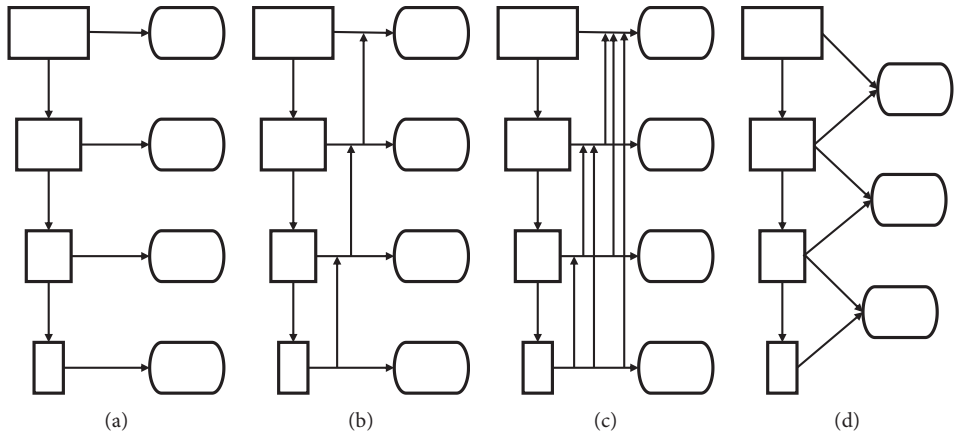

Figure 2: (a) Parallel feature integration architecture of HED [4]; (b) bottom-up feature integration which brings abstract semantics to detailed features [6]; (c) the linear span network proposed by Liu et al. [10] based on linear span theory to enrich the feature fusion; (d) hierarchical feature integration mechanism [8].

through the update of the gradient. Because of their fast realization of searching, usually within 1 GPU day, this kind of method has high hopes and is widely studied but still suffers difficulties to overcome instability.

Search space is a collection of neural networks, which refers to a space that contains all network architectures that could be searched. In order to simplify the search difficulty, the search space is usually limited to a cell space, called cell-based space [15–18]. Cell-based space had been extensively researched and achieved good performances, but it had been criticized for its low difficulty. Some works [19, 20] defined a search space based on the MobileNet [21] block (MBConv), benefiting from effective search and better candidate operations, usually achieving better results than cell-based methods. Thanks to the powerful MBConv, the recent EfficientNet [22], through the introduction of compound coefficient, defines the space about the three dimensions of network length, width, and image resolution and achieved convincing classification results through reinforcement learning.

Some recent methods have applied NAS not only on classification tasks but also to downstream tasks such as object detection and semantic segmentation and have also achieved state-of-the-art performances. In order to adaptively obtain the multiscale side-output fusion structure, NAS-FPN [23] defines a search space, in which the hierarchical feature fusion can be optimized. Under reinforcement learning, NAS-FPN achieves SOTA detection results. Auto-DeepLab [24] greatly improved the performance of segmentation through searching a well-designed search space, including the most popular hand-designed networks, e.g., U-net [25] and hourglass [26].

In this paper, we propose a genetic feature fusion (GFF) method. Based on VGG, we allow each convolutional layer

to accept inputs from multiple earlier convolutional layers, in this way, to achieve multiscale feature fusion, and to maximize the use of pretrained parameters. The network structure of GFF includes an encoder network and a decoder network. The inputs of each layer of the encoder can only come from the encoder, and the inputs of the decoder can come from both encoder and decoder. To reduce the complexity, we limit the number of inputs that can be used.

3. Genetic Feature Fusion

3.1. Revisiting Architecture Search Strategy. Enumerating all the architectures from the search space is the most straightforward strategy to search for the best network. Nevertheless, it has to deal with too enormous computational overhead to be acceptable, and heuristic methods become indispensable. The main heuristic methods of NAS are as follows: reinforcement learning, generic algorithm/evolutionary algorithm, and differentiable search. All these three strategies can achieve good results in a specific search space, so researchers turned to the heuristic search strategies. The basic framework is shown in Figure 3.

3.1.1. Reinforcement Learning. NASNet [16] first introduced reinforcement learning to the NAS community. Reinforcement learning-based methods treat architecture search as an agent learning process and predict good architectures by training the agent. Each network architecture can be represented by a character string, which is generated by an RNN controller. The strings are independently trained from scratch, and the correctness rate is used to train the agent. The agent then instructs the RNN controller to update its parameters so that better architectures can be probably generated next time. Neural architecture search can be performed by looping the above process. However, reinforcement learning-based NAS methods require lots of computational overhead. To predict good network architectures, they need to train a large number of generated architectures. For example, the search of NASNet [16] takes 1800 GPU days.

3.1.2. Genetic Algorithm. Genetic algorithms are inspired by the biological evolution process in nature, which retain superior individuals and discard inferior individuals through continuous iteration of populations that contain all the individuals. Genetic algorithms regard network architectures as individuals and network architecture search as the process of superior individuals gradually emerging from the evolution of the populations. Crossover, mutation, and selection are common operations of genetic algorithms. The crossover occurs between two individuals, each corresponding to its unique coded genes, and the two individuals exchange part of their genes with a predefined probability. The mutation is directed at a single individual, and it mutates with a certain probability to some genes corresponding to that one. Firstly, all the individuals in the population crossover one by one, then each individual is mutated once, and all the new individuals form the new population. After training all the individuals in the populations independently

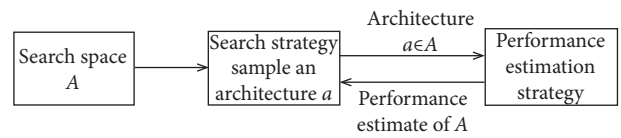

Figure 3: NAS framework consists of three core elements: search space, search strategy, and evaluation strategy.

from scratch, the selection of the superior individuals by certain criteria, such as accuracy rate, is called selection operation. Early genetic algorithm-based methods still require independent training of each network architecture from scratch, so these methods also require high computational costs, such as the search of AmoebaNet [14] which took 3150 GPU days. However, some of the recent methods have achieved state-of-the-art performance with fewer iterations and computation time [27, 28].

3.1.3. Differentiable Search. Reinforcement learning and genetic algorithm-based methods are slow because the evaluation phase takes lots of computational time; that is, every sampled network architecture is trained from scratch. To alleviate this issue, many methods have been proposed, such as using fewer parameters and smaller networks or searching on smaller proxy datasets, but these attempts simply bypass the computational overhead at the expense of a big gap, but the problem remains. Later, a weight-sharing mechanism is proposed to encode the entire search space into a supernetwork [15, 17, 29], in which each searched architecture corresponds to a subarchitecture. The modules in the supernetwork can be shared by all different subarchitectures so that the search procedure is largely accelerated. Differentiable architecture search strategy relaxes the weight-sharing supernetwork space to make it continuous by adding a weight for each candidate operation. DARTS [15], one of the most popular differentiable search algorithms, defined a cell-based search space, which initializes a weight $1/C$ of C candidate operations of each node. Then, these weights can be updated in a continuous space by gradient optimization. The procedure of updating the weights corresponding to these candidate operations is the procedure of searching architecture. At the end of searching, operations with higher weights are retained, and operations with lower weights are discarded, that is, a discretization process is performed.

3.2. Architecture Space for Feature Fusion. The overall network structure has two parts: encoder and decoder, both of which are VGG networks where the decoder network is equivalent to the reversed VGG network structure, except that the pooling layers in the decoder are all replaced by upsampling operations (Figure 4). Including the pooling layers, VGG-16 has 18 layers in all, which are labelled 1–18 sequentially. It is worth to note that the deeper the layer is in the encoder, the larger the number is. For example, the 7th layer in the encoder is after the 6th layer, so the deeper layer's number is larger, while the deeper the layer is in the decoder, the smaller the number is. For example, the 6th decoder layer

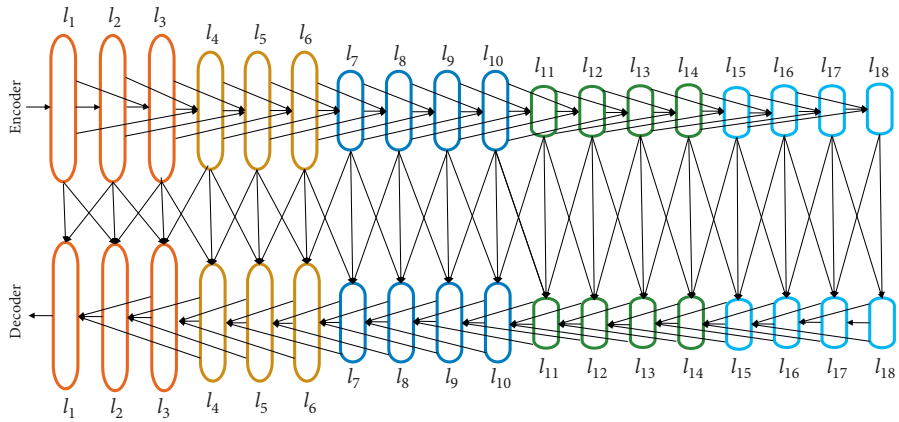

Figure 4: Diagram of the expanded search space based on the VGG [9] network.

is after the 7th decoder layer, that is, the 6th layer is deeper and closer to the first decoder layer.

The search scope is the input of each convolution layer in the encoder and decoder, that is, where the input of each layer comes from. We make a constrain that the input of each convolution layer in the encoder can only come from the encoder, and each convolution layer in the decoder can come from both decoder and the encoder. In order to simplify the space, the range of the candidate input of each convolution layer in the encoder is constrained to three, that is to say, each convolution layer can only accept the output of the three layers before as its input. For example, the 5th layer can only accept the outputs of the second, third, and/or fourth layers. As the decoder can accept the input from both the encoder and the decoder itself, the range of candidate inputs for each layer (convolution or pooling) in the decoder is specified to be 6, where the upper range of the decoder layer is 3, and the encoder layer is 3. The examples of acceptable inputs for each decoder layer are as follows: the 6th layer in the decoder network can accept the 9th, 8th, and 7th decoder layers and the 5th, 6th, and 7th encoder layers.

3.3. Genetic Architecture Search Algorithm. The search algorithm is similar to AdaLSN [27]. As described above, each layer in the network is numbered differently for the indexing purpose. The input of each layer can be represented by a simple list, called the genome, in which each element, called a gene, indicates whether the candidate input layer is selected as the input so that a network individual in the search space can be uniquely encoded as a list set, called a genotype. We specify that each layer accepts the first three layers as inputs and forces the previous layer to be retained to ensure that the entire VGG network is retained. So, except for the marginal layer, there are two genes in each genome on the encoder and five genes in each genome on the decoder.

We randomly initialize 24 individuals to form the initial population. Each individual is trained for 1000 steps from scratch, which is enough for a network to be trained well since most of the parameters are retained. Then, the evaluation process is performed on the validation set that contains 50 images separated from the training set. The eight individuals with the lowest losses are selected as the dominant ones.

Crossover process occurs on the eight selected dominant individuals, paired in pairs, and candidate inputs of each layer, a genome, are exchanged with probability so that 8 new individuals are generated. Each gene of each new individual is mutated with probability p_m, resulting in 8 new individuals forming a new population. Figure 5 shows the overall process. The preceding process is executed for 50 times, and then the final architecture is obtained.

The final network structure will be followed by a convolution to reduce the number of channels to 1, crop the gray image to the size of the original image, and then supervise its distance from the ground truth through the commonly used binary cross-entropy.

4. Experiments

4.1. Experimental Setting

4.1.1. Datasets. We trained and tested our method on 3 datasets which are commonly used for object skeleton detection: SK-LARGE [30], SK506 [5], and WH-SYMMAX [31]. SK-LARGE is sampled from the MS-COCO dataset [32], which contains 746/745 images for training and testing, with 16 classes of objects included. SK506, also called SK-SMALL, is the old version of SK-LARGE, which contains 300 training images and 206 test images in 16 different objects. WH-SYMMAX contains 200/100 training and test images all about Weizmann Horse. All training images and ground truths are augmented with scaling (0.8x, 1.0x, and 1.2x), rotating (0°, 90°, 180°, and 270°), and flipping (right to left and left to right), as well as resolution normalization [33].

4.1.2. Implementation. The search process is executed on NVIDIA TITAN RTX GPUs (24 GB of memory). VGG [9] is used as the backbone network for evolution. The crossover rate p_e and mutation rate p_m are both 0.2 and 50 generations in total. One final searched architecture is shown in Figure 6.

The final architecture was trained and tested on single NVIDIA GTX 2080Ti (12 GB of memory). We used the

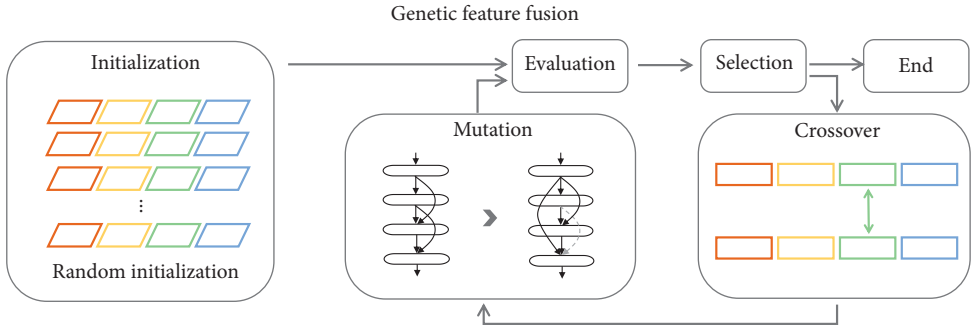

Figure 5: Framework of our GFF. The initial population is evaluated, which adopts the random initialization, and dominant individuals perform crossover and mutation to form a new population, and then the above process is iterated.

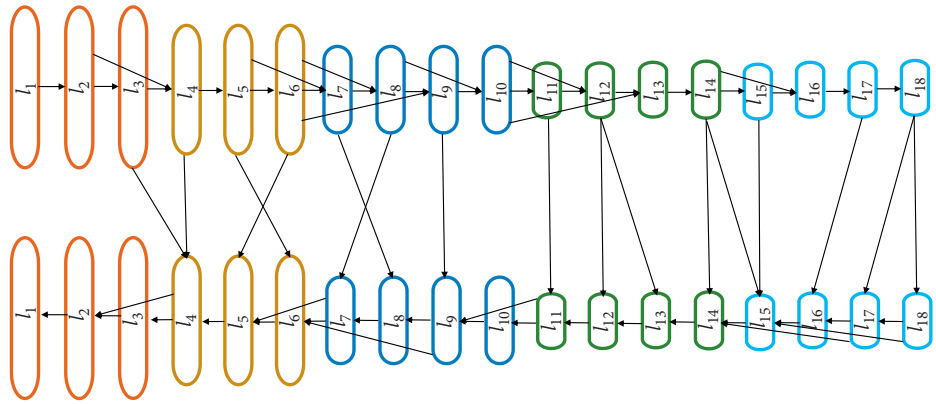

Figure 6: Diagram of one searched architecture.

Adam optimizer [34] with the initial learning rate of $1e{-}7$, weight decay of 0.0002, and betas of (0.9, 0.999). During training, if batch size is 1, the network parameter would be updated using the added gradient calculated every 10 iterations of forwarding propagation. The learning rate is fixed during the first 50,000 iterations and decreased to one-tenth of the original. The training process would be stopped in iteration 60,000. The parameters of the searched architecture are about 32.1 M, with flops of 72.6 G when inputting a 224×224 image, and the training process took 9.5 hours.

The setting in [35] is used as the evaluation metrics.

4.2. Performance and Comparison. Our GFF outperforms the state of the art of object skeleton detection in several commonly used datasets, and the results are shown in Table 1. VGG [9] with pretrained parameters is used for our proposed method, and the searched architecture is retrained on each dataset.

The skeleton results are shown and compared with other methods in Figure 7. One can see the superiority of our GFF, which can extract skeleton maps of different granularities with better continuity.

On the SK-LARGE [30] dataset, our GFF achieves the performance of F-score 73.6%, which is the highest object skeleton detection performance compared to the optimal method DeepFlux [37], which achieves 73.2%. We also outperform Hi-Fi [8], which utilizes the additional scale-associated ground truth with a margin of 1.2%. In another convincing dataset SK506 [5], GFF also embodies superiority. GFF achieves the F-score of 72.3%, which is higher than Hi-Fi [8] and DeepFlux [37] with margins of 4.2% and 2.3% separately.

4.3. Ablation Study. We verify the effectiveness of our method through comparative experiments which perform search process randomly. We randomly reserve each candidate operation for each layer and execute 4 times independently to get 4 architectures. Each architecture is retrained on SK-LARGE for the same number of steps, and the results are shown in Table 2. One can find that the results of these 4 architectures range from 70.9% to 72.7%, which is lower than 73.6% achieved by the architecture we searched.

Figure 8 shows the adaptiveness of the GFF during the search process. Figure 8(a) shows the change in the loss of top1–4 in different generations. The values of loss are obviously decreasing, which indicates that better architectures appear in the population as it evolves. Figure 8(b) shows the change in the number of candidate input connections reserved on the encoder and the decoder in different generations. In general, the number of candidate connections is decreasing, which indicates that our method is adaptively searching for scale-aware architectures. Figure 8(c) shows

Table 1: Performance comparison of the state-of-the-art approaches on commonly used skeleton\symmetry detection datasets.

Methods	Datasets		
	SK-LARGE [30]	SK-SMALL [5]	WH-SYMMAX [31]
MIL [35]	0.353	0.392	0.365
HED [4]	0.497	0.541	0.732
RCF [36]	0.626	0.613	0.751
FSDS [5]	0.633	0.623	0.769
SRN [6]	0.658	0.632	0.443
LSN [10]	0.668	0.633	0.797
Hi-Fi [8]	0.724	0.681	0.805
DeepFlux [37]	0.732	0.695	0.840
GFF (ours)	0.736	0.723	0.837

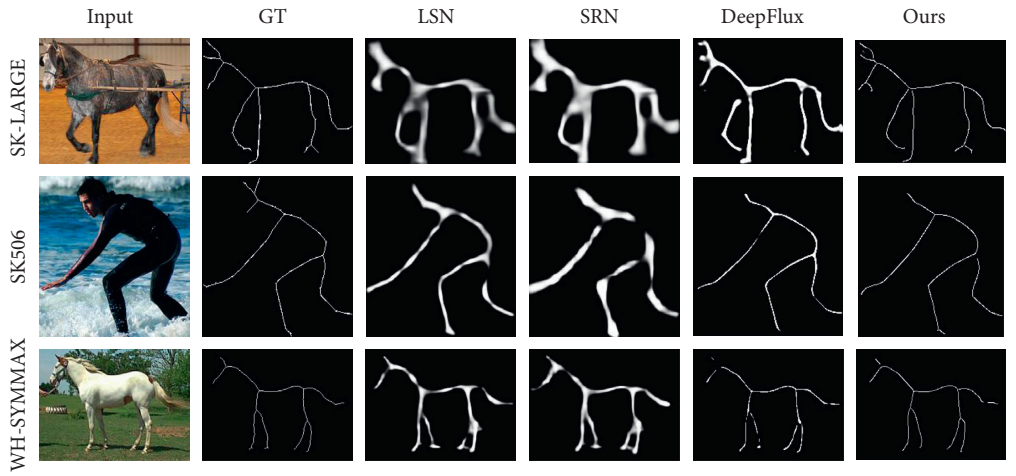

Figure 7: Skeleton detection examples by state-of-the-art approaches on SK-LARGE [30], SK-SMALL [5], and WH-SYMMAX [31].

Table 2: Performance of random search.

Archi1	Archi2	Arch31	Archi4
0.714	0.723	0.709	0.727

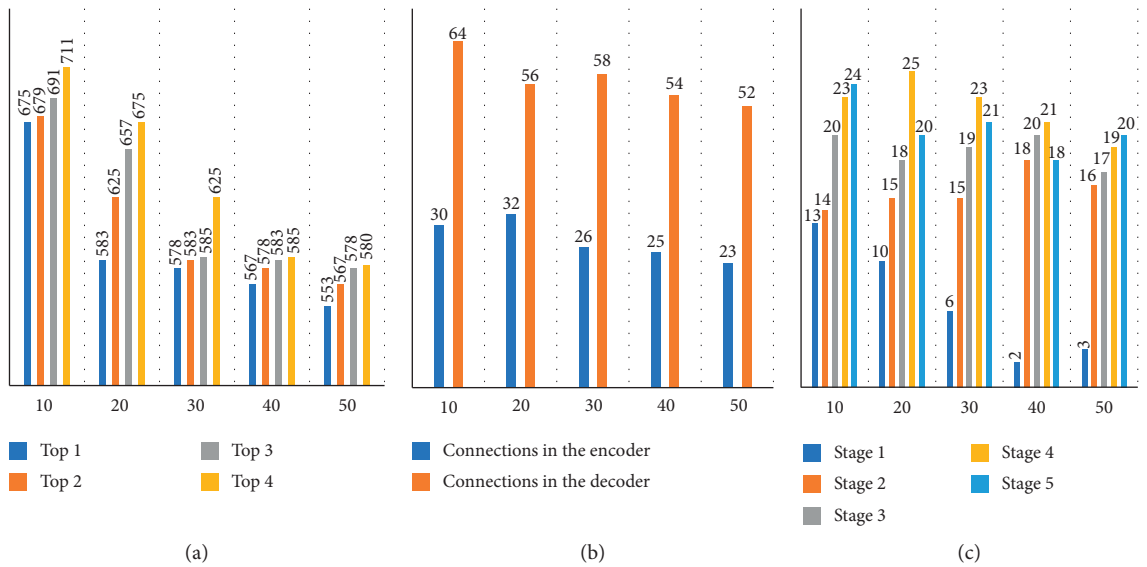

Figure 8: The horizontal coordinate of all figures indicates the generations. (a) Changes of loss of top1–4 architectures. (b) Changes in the number of connections reserved for the encoder and decoder. (c) Changes in the number of connections reserved for stages 1–5 in the framework.

Table 3: Other models.

VGG	FHN [38]
0.480	0.675

the number of candidate input connections reserved in different stages. One can find that the connections of all stages except stage 2 are decreasing because the features of stage 2 are well-balanced, fine-grained, and coarse-grained features, but other stages reserve fewer and fewer candidate input connections, which makes sense of our method's search process.

To demonstrate the superiority of our GFF, in Table 3, we test VGG which achieves the F-score of 48.0% that is 25.6% lower than ours. FHN [38], with similar motivation, is 6.1% lower than our GFF.

5. Conclusion

Object skeleton detection has aroused widespread research interest in the field of computer vision, helping to understand the attributes of shape and topology of objects in natural images. In this paper, we proposed genetic feature fusion (GFF), which expanded the existing classification networks to form a network space set. Each network, including an encoder and decoder, in this space could make full use of pretrained parameters. Driven by neural architecture search (NAS), GFF automatically integrates features of different scales and searches the scale-aware networks for object skeleton detection in the expanded space. The significant higher performance in commonly used datasets demonstrated the superiority of GFF.

Data Availability

The dataset used in our paper can be made available at https://kaizhao.net/sk-large https://openaccess.thecvf.com/content_cvpr_2016/html/Shen_Object_Skeleton_Extraction_CVPR_2016_paper.html https://dl.acm.org/doi/10.1016/j.patcog.2015.10.015.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported in part by the National Natural Science Foundation of China (NSFC) (Grant nos. 61836012 and 61771447) and Strategic Priority Research Programme of Chinese Academy of Sciences (Grant no. XDA27010303).

References

[1] C. L. Teo, C. Fermüller, and Y. Aloimonos, "Detection and segmentation of 2d curved reflection symmetric structures," in *Proceedings of the 2015 IEEE International Conference on Computer Vision*, pp. 1644–1652, Santiago, Chile, December 2015.

[2] S. Wei, V. Ramakrishna, T. Kanade, and Y. Sheikh, "Convolutional pose machines," in *Proceedings of the*, pp. 4724–4732proceeding 2016 IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, June 2016.

[3] Z. Zhang, W. Shen, C. Yao, and X. Bai, "Symmetry-based text line detection in natural scenes," in *Proceedings of the*, pp. 2558–2567proceeding 2015 IEEE Conference on Computer Vision and Pattern Recognition, Boston, MA, USA, June 2015.

[4] S. Xie and Z. Tu, "Holistically-nested edge detection," in *Proceedings of the 2015 IEEE International Conference on Computer Vision*, pp. 1395–1403, Santiago, Chile, December 2015.

[5] W. Shen, K. Zhao, Y. Jiang, Y. Wang, Z. Zhang, and X. Bai, "Object skeleton extraction in natural images by fusing scale-associated deep side outputs," in *Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 222–230, Las Vegas, NV, USA, June 2016.

[6] W. Ke, J. Chen, J. Jiao, G. Zhao, and Q. Ye, "SRN: side-output residual network for object symmetry detection in the wild," in *Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 302–310, Honolulu, HI, USA, July 2017.

[7] C. Liu, W. Ke, J. Jiao, and Q. Ye, "RSRN: rich side-output residual network for medial axis detection," in *Proceedingsof the 2017 IEEE International Conference on Computer Vision Workshops*, pp. 1739–1743, Venice, Italy, October 2017.

[8] K. Zhao, W. Shen, S. Gao, D. Li, and M. Cheng, "Hi-fi: Hierarchical feature integration for skeleton detection," in *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence {IJCAI-18}*, pp. 1191–1197, Stockholm, Sweden, July 2018.

[9] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," 2015, http://arxiv.org/abs/1409.1556v6.

[10] C. Liu, W. Ke, F. Qin, and Q. Ye, "Linear span network for object skeleton detection," in *Proceedings of the 2008 15th European Conference on Computer Vision*, pp. 133–148, Munich, Germany, September 2018.

[11] C. Liu, F. Wan, X. Zhang, Y. Yao, W. Ke, and Q. Ye, "Orthogonal decomposition network for pixel-wise binary classification," in *Proceedings of the 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 6064–6073, Long Beach, CA, USA, June 2019.

[12] C. Liu, D. Luo, Y. Zhang, W. Ke, F. Wan, and Q. Ye, "Parametric skeleton generation via Gaussian mixture models," in *Proceedings of the 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pp. 1167–1171, Long Beach, CA, USA, June 2019.

[13] B. Zoph and Q. V. Le, "Neural architecture search with reinforcement learning," 2017.

[14] E. Real, A. Aggarwal, Y. Huang, and Q. V. Le, "Regularized evolution for image classifier architecture search," in *Proceedings of the Thirty-Third AAAI Conference on artificial intelligence, AAAI 2019*, Honolulu, HI, USA, January 2019.

[15] H. Liu, K. Simonyan, and Y. Yang, "DARTS: differentiable architecture search," 2019, http://arxiv.org/abs/1806.09055v2.

[16] B. Zoph, V. Vasudevan, J. Shlens, and Q. V. Le, "Learning transferable architectures for scalable image recognition," in *Proceedings of the 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 8697–8710, Salt Lake City, UT, USA, June 2018.

[17] Y. Xu, L. Xie, X. Zhang et al., "PC-DARTS: partial channel connections for memory-efficient differentiable architecture search," 2020, http://arxiv.org/abs/1907.05737.

[18] Y. Tian, C. Liu, L. Xie, J. Jiao, and Q. Ye, "Discretization-aware architecture search," 2020, https://arxiv.org/abs/2007.03154.

[19] M. Tan, B. Chen, R. Pang et al., "Mnasnet: platform-aware neural architecture search for mobile," in *Proceedings of the 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Long Beach, CA, USA, June 2019.

[20] J. Fang, Y. Sun, Q. Zhang, Y. Li, W. Liu, and X. Wang, "Densely connected search space for more flexible neural architecture search," in *Proceedings of the 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, Seattle, WA, USA, June 2020.

[21] A. G. Howard, M. Zhu, B. Chen et al., "Efficient convolutional neural networks for mobile vision applications," 2017, https://arxiv.org/abs/1704.04861v1.

[22] T. Mingxing and Q. V. Le, "Efficientnet: rethinking model scaling for convolutional neural networks," in *Proceedings of the 2019 36th International Conference on Machine Learning*, Long Beach, CA, USA, June 2019.

[23] G. Ghiasi, T. Lin, and Q. V. Le, "NAS-FPN: learning scalable feature pyramid architecture for object detection," in *Proceedings of the 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 7036–7045, Long Beach, CA, USA, June 2019.

[24] C. Liu, L. Chen, F. Schroff et al., "Auto-deeplab: hierarchical neural architecture search for semantic image segmentation," in *Proceedings of the 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 82–92, Long Beach, CA, USA, June 2019.

[25] O. Ronneberger, P. Fischer, and T. Brox, "U-net: convolutional networks for biomedical image segmentation," in *Proceedings of the 2015 18th International Conference*, Munich, Germany, October 2015.

[26] A. Newell, K. Yang, and J. Deng, "Stacked hourglass networks for human pose estimation," in *Proceedings of the Computer Vision-ECCV 2016 14th European Conference*, Amsterdam, The Netherlands, October 2016.

[27] C. Liu, Y. Tian, J. Jiao, and Q. Ye, "Adaptive linear span network for object skeleton detection," 2020, https://arxiv.org/abs/2011.03972v1.

[28] Y. Chen, Q. Zhang, C. Huang, L. Mu, G. Meng, and X. Wang, "Reinforced evolutionary neural architecture search," 2018, https://arxiv.org/abs/1808.00193.

[29] H. Pham, M. Y. Guan, B. Zoph, Q. V. Le, and J. Dean, "Efficient neural architecture search via parameter sharing," in *Proceedings of the 35th International Conference on Machine Learning, ICML 2018*, Stockholm, Sweden, July 2018.

[30] W. Shen, K. Zhao, Y. Jiang, Y. Wang, X. Bai, and A. L. Yuille, "Deepskeleton: learning multi-task scale-associated deep side outputs for object skeleton extraction in natural images," *IEEE Transactions on Image Processing*, vol. 26, no. 11, 2017.

[31] W. Shen, X. Bai, Z. Hu, and Z. Zhang, "Multiple instance subspace learning via partial random projection tree for local reflection symmetry in natural images," *Pattern Recognition*, vol. 52, pp. 306–316, 2016.

[32] A. Karpathy and L. Fei-Fei, "Deep visual-semantic alignments for generating image descriptions," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, no. 4, pp. 664–676, 2017.

[33] W. Xu, G. Parmar, and Z. Tu, "Geometry-aware end-to-end skeleton detection," 2019.

[34] D. P. Kingma and J. Ba, "Adam: a method for stochastic optimization," in *Proceedings of the 3rd International Conference on Learning Representations*, San Diego, CA, USA, May 2015.

[35] S. Tsogkas and I. Kokkinos, "Learning-based symmetry detection in natural images," in *Proceedings of the 2012 European Conference on Computer Vision*, Florence, Italy, October 2012.

[36] Y. Liu, M. Cheng, X. Hu et al., "Richer convolutional features for edge detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 41, no. 8, pp. 1939–1946, 2019.

[37] Y. Wang, Y. Xu, S. Tsogkas, X. Bai, S. J. Dickinson, and K. Siddiqi, "Deepflux for skeletons in the wild," in *Proceedings of the 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 5287–5296, Long Beach, CA, USA, June 2019.

[38] N. Jiang, Y. Zhang, D. Luo, C. Liu, Y. Zhou, and Z. Han, "Feature hourglass network for skeleton detection," in *Proceedings of the 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, Long Beach, CA, USA, June 2019.