Results of the IEEE CEC 2023 Competition on Multiobjective Neural Architecture Search

Z. Liang, Z. Lu, R. Cheng and Y. Jin

IEEE 2023 Congress on Evolutionary Computation,

Chicago, USA, July 1-5, 2023

content

Motivation

- Neural Architecture Search (NAS) automates deep learning network design, improving performance in computer vision tasks.
- NAS for multiple design criteria is a challenging multiobjective optimization problem (MOP) suitable for evolutionary multiobjective optimization (EMO) algorithms.
- Challenges in NAS include black-box optimization, discrete variables, noisy landscapes, and many objectives, etc.
- There is still No tailored NAS test suites available.

Competition Entries

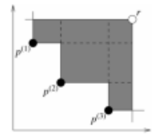
- 10 entries： five new algorithms and five baseline algorithms
- The baseline algorithms were run on PlatEMO[2].

TABLE III: The information of competition entries

ID	Algorithm	Authors	Description	Label
1	CMOSMA_NCHU	Chao He et al.	A Two Population Evolutionary Framework for Handling NAS Problems	Participant
2	DLEA-Niche	Gui Li, Guining Zhan	Dynamic Learning Evolutionary Algorithm with Niche-based Diversity Maintenance Strategy	Participant
3	EABSM-NAS	Chixin Wang et al.	Evolutionary Algorithm Based on Surrogate Models in Neural Architecture Search	Participant
4	NSGA2-DER	Pengcheng Jiang, Chenchen Zhu	Improved Evolutionary Operators Based on Different Encoding Regions	Participant
5	Regional NSGAII	Xujia Zhang	Regional NSGAII	Participant
6	HypE[3]	Johannes Bader, Eckart Zitzler	An Algorithm for Fast Hypervolume-Based Many-Objective Optimization	Baseline
7	KnEA[4]	Xingyi Zhang et al.	A Knee Point-Driven Evolutionary Algorithm for Many-Objective Optimization	Baseline
8	MOEA/D[5]	Qingfu Zhang, Hui Li	A Multiobjective Evolutionary Algorithm Based on Decomposition	Baseline
9	NSGA-II[6]	Kalyanmoy Deb et al.	A Fast and Elitist Multiobjective Genetic Algorithm	Baseline
10	RVEA[7]	Ran Cheng et al.	A Reference Vector Guided Evolutionary Algorithm for Many-Objective Optimization	Baseline

Performance Indicators

- Hypervolume (HV)[8]



The HV in the two-objective case[8]

- Inverted generational distance (IGD)[9]

$$\mathrm{IGD} = \frac{\sum_{x \in P^*} \min_{y \in P} dis(x, y)}{|P^*|}$$

- The performance indicator that calculates IGD and HV are from **pymoo**[10]

Conclusion

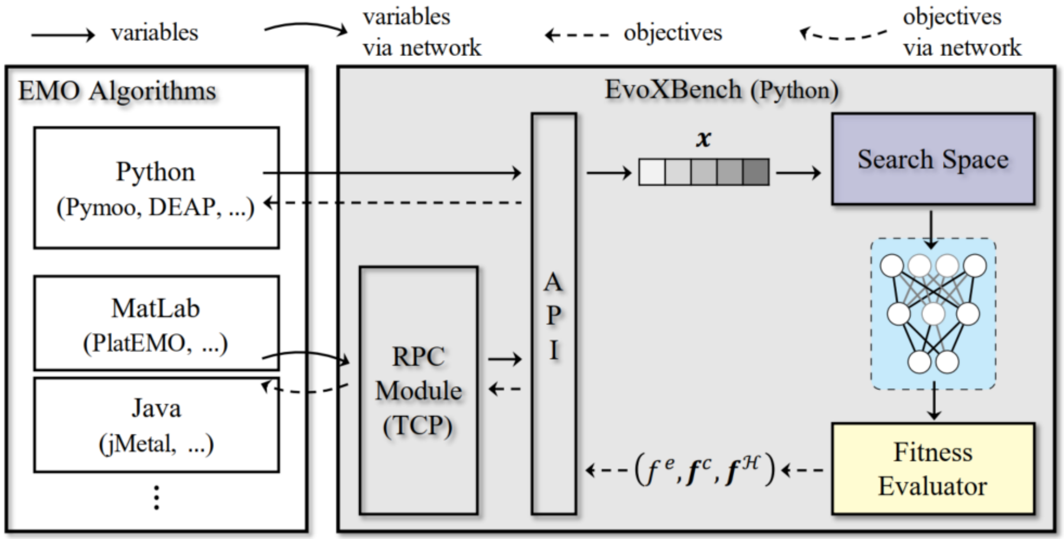
- According to the final ranking based on HV metrics, the winner is CMOSMA_NCHU(A two population evolutionary framework for handling NAS problems)
- The competition showcased the effectiveness of EMO algorithms in addressing multiobjective NAS tasks.
- EMO algorithms offer a promising approach for automated network design in complex application scenarios such as autonomous driving.

Motivation

- Neural Architecture Search (NAS) automates deep learning network design, improving performance in computer vision tasks.
- NAS for multiple design criteria is a challenging multiobjective optimization problem (MOP) suitable for evolutionary multiobjective optimization (EMO) algorithms.
- Challenges in NAS include black-box optimization, discrete variables, noisy landscapes, and many objectives, etc.
- There is still No tailored NAS test suites available.

EvoXBench

- Support multiple programming languages
- Run without the requirement of GPUs or Pytorch/Tensorflow
- Complete image classification performance prediction for 1,000 different types of DNN architectures in 0.1 seconds



There are two representative benchmark test suites in EvoXBench[1], i.e., the C-10/MOP(tailored for image classification on CIFAR-10) and the IN-1K/MOP(tailored for image classification on ImageNet 1K).

Test Suites

TABLE I: Definition of the proposed C-10/MOP test suite.

Problem	Ω	D	M	Objectives
C-10/MOP1	NB101	26	2	f^e, f_1^c
C-10/MOP2	NB101	26	3	f^e, f_1^c, f_2^c
C-10/MOP3	NATS	5	3	f^e, f_1^c, f_2^c
C-10/MOP4	NATS	5	4	f^e, f_1^c, f_2^c, $f_1^{h_1}$
C-10/MOP5	NB201	6	5	f^e, f_1^c, f_2^c, $f_1^{h_1}$, $f_2^{h_1}$
C-10/MOP6	NB201	6	6	f^e, f_1^c, f_2^c, $f_1^{h_2}$, $f_2^{h_2}$, $f_3^{h_2}$
C-10/MOP7	NB201	6	8	f^e, f_1^c, f_2^c, $f_1^{h_1}$, $f_2^{h_1}$, $f_1^{h_2}$, $f_2^{h_2}$, $f_3^{h_2}$
C-10/MOP8	DARTS	32	2	$^\dagger f^e$, f_1^c
C-10/MOP9	DARTS	32	3	$^\dagger f^e$, f_1^c, f_2^c

† indicates that f^e (validation error) is based on surrogate modeling.
‡ hardware h_1 = GPU and h_2 = Eyeriss.

TABLE II: Definition of the proposed IN-1K/MOP test suite.

Problem	Ω	D	M	Objectives
IN-1K/MOP1	ResNet50	25	2	f^e, f_1^c
IN-1K/MOP2	ResNet50	25	2	f^e, f_2^c
IN-1K/MOP3	ResNet50	25	3	f^e, f_1^c, f_2^c
IN-1K/MOP4	Transformer	34	2	f^e, f_1^c
IN-1K/MOP5	Transformer	34	2	f^e, f_2^c
IN-1K/MOP6	Transformer	34	3	f^e, f_1^c, f_2^c
IN-1K/MOP7	MNV3	21	2	f^e, f_1^c
IN-1K/MOP8	MNV3	21	3	f^e, f_1^c, f_2^c
IN-1K/MOP9	MNV3	21	4	f^e, f_1^c, f_2^c, $f_1^{h_1}$

† All f^e (validation errors) are based on surrogate modeling.
‡ hardware h_1 = Samsung Note10.

Notations:

f^e: prediction error

f_1^c: number of parameters (M)

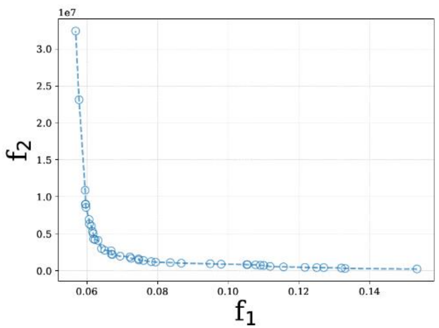
f_2^c: number of FLOPs (M)

$f_1^{h_1}, f_1^{h_2}$: latency (ms)

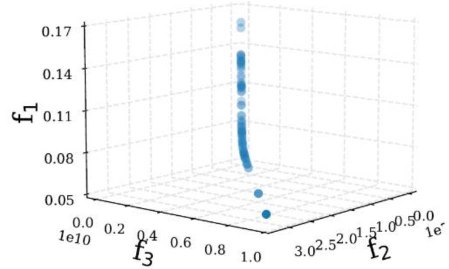
$f_2^{h_1}, f_2^{h_2}$: energy (mJ)

$f_3^{h_2}$: Arithmetic Intensity (ops/byte)

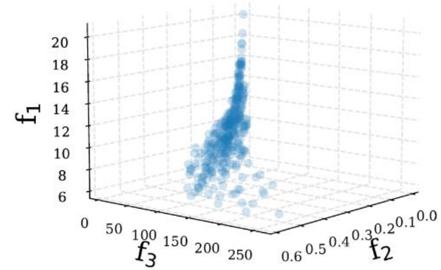
Pareto Front of C-10/MOP1 - C-10/MOP7



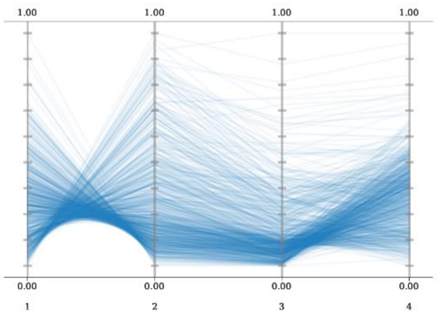
(a) C-10/MOP1



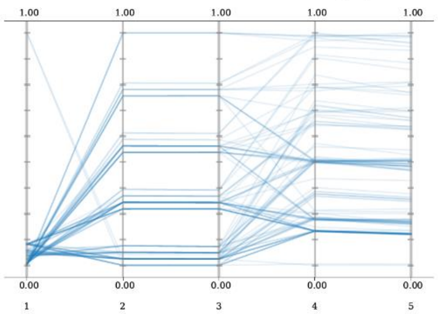
(b) C-10/MOP2



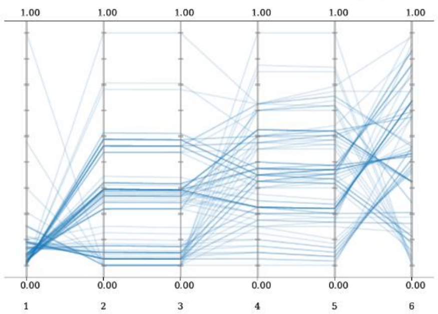
(c) C-10/MOP3



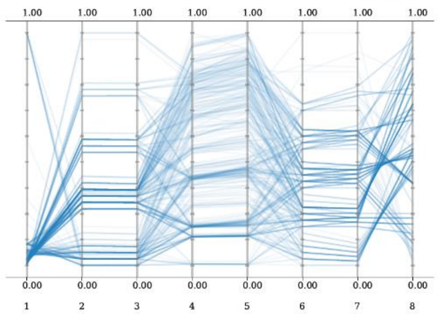
(d) C-10/MOP4



(e) C-10/MOP5



(f) C-10/MOP6



(g) C-10/MOP7

Competition Entries

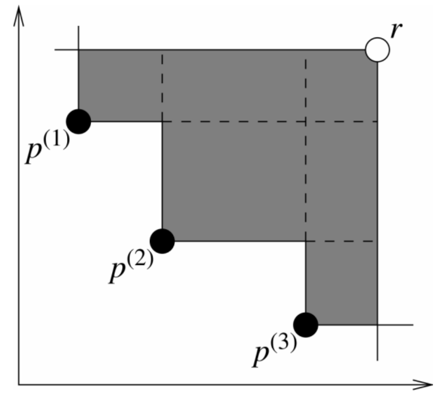
- 10 entries： five new algorithms and five baseline algorithms
- The baseline algorithms were run on PlatEMO[2].

TABLE III: The information of competition entries

ID	Algorithm	Authors	Description	Label
1	CMOSMA_NCHU	Chao He et al.	A Two Population Evolutionary Framework for Handling NAS Problems	Participant
2	DLEA-Niche	Gui Li, Guining Zhan	Dynamic Learning Evolutionary Algorithm with Niche-based Diversity Maintenance Strategy	Participant
3	EABSM-NAS	Chixin Wang et al.	Evolutionary Algorithm Based on Surrogate Models in Neural Architecture Search	Participant
4	NSGA2-DER	Pengcheng Jiang, Chenchen Zhu	Improved Evolutionary Operators Based on Different Encoding Regions	Participant
5	Regional NSGAII	Xujia Zhang	Regional NSGAII	Participant
6	HypE[3]	Johannes Bader, Eckart Zitzler	An Algorithm for Fast Hypervolume-Based Many-Objective Optimization	Baseline
7	KnEA[4]	Xingyi Zhang et al.	A Knee Point-Driven Evolutionary Algorithm for Many-Objective Optimization	Baseline
8	MOEA/D[5]	Qingfu Zhang, Hui Li	A Multiobjective Evolutionary Algorithm Based on Decomposition	Baseline
9	NSGA-II[6]	Kalyanmoy Deb et al.	A Fast and Elitist Multiobjective Genetic Algorithm	Baseline
10	RVEA[7]	Ran Cheng et al.	A Reference Vector Guided Evolutionary Algorithm for Many-Objective Optimization	Baseline

Performance Indicators

- Hypervolume (HV)[8]



The HV in the two-objective case[8]

- Inverted generational distance (IGD)[9]

$$\mathrm{IGD} = \frac{\sum_{x \in P^*} \min_{y \in P} dis(x, \ y)}{|P^*|}$$

- The performance indicator that calculates IGD and HV are from **pymoo**[10]

Ranking Strategy

- For each problem, each algorithm is executed 31 times and the HV is calculated.
- The mean value and standard deviation of HV is calculated for each problem and sorted.
- The statistical results are compared using Wilcoxon rank sum test.
- The score for each algorithm is determined by summing up its ranks across all problems and then normalizing the result.

Overview of the Results

TABLE IV: Statistical results (mean and standard deviation) of the HV values on C-10/MOP test suite. The best results of each instance are in bold red.

	CMOSMA_NCHU	DLEA-Niche	EABSM-NAS	HypE	KnEA	MOEA/D	NSGA-II	NSGA2-DER	Regional NSGAII	RVEA
C-10/MOP1	**0.9434 (0.0069)≈**	0.9377 (0.0061)-	**0.9439 (0.0024)≈**	0.8060 (0.1129)-	0.8927 (0.0123)-	0.8965 (0.0107)-	0.9156 (0.0094)-	0.8984 (0.0134)-	0.9373 (0.0038)-	0.9158 (0.0100)-
C-10/MOP2	0.9179 (0.0027)-	0.9165 (0.0046)-	**0.9235 (0.0013)+**	0.7714 (0.0817)-	0.8844 (0.0121)-	0.8714 (0.0088)-	0.8987 (0.0075)-	0.8756 (0.0127)-	0.9113 (0.0023)-	0.8903 (0.0087)-
C-10/MOP3	0.8272 (0.0035)-	**0.8288 (0.0040)+**	0.8236 (0.0021)-	0.7924 (0.0065)-	0.7986 (0.0142)-	0.7883 (0.0039)-	0.8054 (0.0030)-	0.8102 (0.0023)-	0.8069 (0.0018)-	0.8013 (0.0059)-
C-10/MOP4	**0.7781 (0.0075)≈**	**0.7811 (0.0058)≈**	0.7740 (0.0032)-	0.7483 (0.0099)-	0.7428 (0.0122)-	0.7231 (0.0134)-	0.7452 (0.0075)-	0.7592 (0.0066)-	0.7594 (0.0095)-	0.7419 (0.0173)-
C-10/MOP5	**0.7127 (0.0001)≈**	0.7124 (0.0003)-	**0.7127 (0.0001)≈**	0.7107 (0.0004)-	0.7109 (0.0003)-	0.6975 (0.0035)-	0.7110 (0.0004)-	0.7081 (0.0011)-	0.7107 (0.0004)-	0.6984 (0.0157)-
C-10/MOP6	**0.7404 (0.0001)+**	0.7402 (0.0005)-	0.7370 (0.0027)-	0.7367 (0.0006)-	0.7370 (0.0008)-	0.7101 (0.0108)-	0.7377 (0.0003)-	0.7319 (0.0045)-	0.7379 (0.0003)-	0.7151 (0.0086)-
C-10/MOP7	0.5793 (0.0095)-	**0.5839 (0.0041)≈**	0.5714 (0.0121)-	**0.5844 (0.0053)≈**	0.5784 (0.0085)-	0.5403 (0.0301)-	0.5751 (0.0091)-	0.5746 (0.0108)-	0.5780 (0.0053)-	0.5493 (0.0174)-
C-10/MOP8	**0.9776 (0.0017)+**	0.9709 (0.0049)-	0.9757 (0.0033)-	0.9398 (0.0114)-	0.9463 (0.0050)-	0.8788 (0.0258)-	0.9462 (0.0067)-	0.9333 (0.0105)-	0.9511 (0.0037)-	0.9116 (0.0193)-
C-10/MOP9	**0.9623 (0.0012)+**	0.9523 (0.0137)-	0.9607 (0.0046)-	0.9271 (0.0087)-	0.9310 (0.0065)-	0.8226 (0.0288)-	0.9293 (0.0113)-	0.9025 (0.0182)-	0.9324 (0.0034)-	0.8999 (0.0155)-

+ indicates a method achieving significantly better performance.
≈ indicates a method achieving similar performance as the best-performing method.
− indicates a method achieving significantly worse performance.

Overview of the Results

TABLE V: Statistical results (mean and standard deviation) of the HV values on IN-1K/MOP test suite. The best results of each instance are in bold red.

	CMOSMA_NCHU	DLEA-Niche	EABSM-NAS	HypE	KnEA	MOEA/D	NSGA-II	NSGA2-DER	Regional NSGAII	RVEA
IN-1K/MOP1	0.9261 (0.0044)-	0.9092 (0.0074)-	**0.9327 (0.0031)+**	0.9029 (0.0087)-	0.8762 (0.0088)-	0.8622 (0.0111)-	0.9040 (0.0076)-	0.8866 (0.0051)-	0.9113 (0.0042)-	0.7776 (0.0446)-
IN-1K/MOP2	0.8853 (0.0012)-	0.8790 (0.0019)-	**0.8867 (0.0007)+**	0.8592 (0.0050)-	0.8376 (0.0069)-	0.8195 (0.0053)-	0.8687 (0.0038)-	0.8393 (0.0067)-	0.8633 (0.0029)-	0.2710 (0.1459)-
IN-1K/MOP3	**0.8115 (0.0048)≈**	**0.8095 (0.0057)≈**	0.8050 (0.0032)-	0.8069 (0.0042)-	0.7729 (0.0145)-	0.7074 (0.0136)-	0.7940 (0.0059)-	0.7901 (0.0057)-	0.7955 (0.0040)-	0.7401 (0.0149)-
IN-1K/MOP4	**0.9938 (0.0053)≈**	0.9802 (0.0060)-	**0.9947 (0.0036)≈**	0.9307 (0.0290)-	0.9108 (0.0443)-	0.7867 (0.0196)-	0.9346 (0.0171)-	0.9108 (0.0094)-	0.9531 (0.0080)-	0.3743 (0.1063)-
IN-1K/MOP5	**0.9946 (0.0082)≈**	0.9847 (0.0067)-	**0.9978 (0.0045)≈**	0.9330 (0.0231)-	0.9056 (0.0400)-	0.7964 (0.0229)-	0.9439 (0.0258)-	0.9071 (0.0124)-	0.9528 (0.0092)-	0.3907 (0.1238)-
IN-1K/MOP6	**0.9806 (0.0123)≈**	0.9654 (0.0088)-	**0.9845 (0.0042)≈**	0.8951 (0.0267)-	0.9164 (0.0305)-	0.7083 (0.0273)-	0.9210 (0.0199)-	0.8974 (0.0094)-	0.9219 (0.0148)-	0.8341 (0.0217)-
IN-1K/MOP7	0.8954 (0.0163)-	0.8443 (0.0273)-	**0.9198 (0.0019)+**	0.8625 (0.0139)-	0.8432 (0.0193)-	0.7672 (0.0204)-	0.8596 (0.0120)-	0.8375 (0.0072)-	0.8591 (0.0142)-	0.8019 (0.0210)-
IN-1K/MOP8	**0.7094 (0.0065)≈**	**0.7118 (0.0081)≈**	0.6914 (0.0054)-	**0.7085 (0.0103)≈**	0.6257 (0.0328)-	0.4886 (0.0348)-	0.6793 (0.0116)-	0.6547 (0.0092)-	0.6742 (0.0076)-	0.6042 (0.0433)-
IN-1K/MOP9	0.6075 (0.0123)-	**0.6320 (0.0054)≈**	0.6101 (0.0053)-	**0.6313 (0.0067)≈**	0.5260 (0.0301)-	0.3872 (0.0353)-	0.5788 (0.0126)-	0.5748 (0.0074)-	0.5700 (0.0097)-	0.5188 (0.0402)-

\+ indicates a method achieving significantly better performance.
≈ indicates a method achieving similar performance as the best-performing method.
− indicates a method achieving significantly worse performance.

Overview of the Results

TABLE VI: Statistical results (mean and standard deviation) of the IGD values on the first seven instances of C-10/MOP test suite. The best results of each instance are in bold red.

	CMOSMA_NCHU	DLEA-Niche	EABSM-NAS	HypE	KnEA	MOEA/D	NSGA-II	NSGA2-DER	Regional NSGAII	RVEA
C-10/MOP1	**0.0206 (0.0040)≈**	0.0321 (0.0095)-	**0.0206 (0.0027)≈**	0.1306 (0.0415)-	0.0690 (0.0128)-	0.0851 (0.0142)-	0.0498 (0.0138)-	0.0657 (0.0101)-	0.0274 (0.0040)-	0.0546 (0.0119)-
C-10/MOP2	0.0195 (0.0022)-	0.0297 (0.0089)-	**0.0115 (0.0075)+**	0.1245 (0.0393)-	0.0583 (0.0089)-	0.0876 (0.0119)-	0.0453 (0.0102)-	0.0620 (0.0119)-	0.0266 (0.0068)-	0.0586 (0.0081)-
C-10/MOP3	0.0227 (0.0023)-	**0.0213 (0.0027)+**	0.0276 (0.0030)-	0.0478 (0.0040)-	0.0452 (0.0068)-	0.0553 (0.0061)-	0.0386 (0.0021)-	0.0391 (0.0020)-	0.0380 (0.0017)-	0.0427 (0.0042)-
C-10/MOP4	0.0597 (0.0030)-	**0.0535 (0.0019)+**	0.0589 (0.0020)-	0.0727 (0.0040)-	0.0665 (0.0030)-	0.0829 (0.0055)-	0.0697 (0.0029)-	0.0686 (0.0024)-	0.0714 (0.0051)-	0.0768 (0.0057)-
C-10/MOP5	0.0062 (0.0019)-	0.0201 (0.0110)-	**0.0035 (0.0022)+**	0.0911 (0.0214)-	0.0554 (0.0191)-	0.1310 (0.0270)-	0.0465 (0.0079)-	0.0693 (0.0140)-	0.0547 (0.0119)-	0.2359 (0.0486)-
C-10/MOP6	**0.0031 (0.0029)+**	0.0107 (0.0072)-	0.0325 (0.0101)-	0.0609 (0.0157)-	0.0248 (0.0083)-	0.0799 (0.0150)-	0.0147 (0.0054)-	0.0428 (0.0074)-	0.0199 (0.0048)-	0.1184 (0.0217)-
C-10/MOP7	0.0663 (0.0051)-	**0.0332 (0.0054)+**	0.0426 (0.0035)-	0.1441 (0.0164)-	0.0525 (0.0048)-	0.1512 (0.0188)-	0.0589 (0.0045)-	0.0791 (0.0103)-	0.0907 (0.0099)-	0.2118 (0.0266)-

+ indicates a method achieving significantly better performance.
≈ indicates a method achieving similar performance as the best-performing method.
− indicates a method achieving significantly worse performance.
(IGD value is not used to calculate score when ranking)

Overview of the Ranks

TABLE VII: Ranks according to HV values

	CMOSMA_NCHU	DLEA-Niche	EABSM-NAS	HypE	KnEA	MOEA/D	NSGA-II	NSGA2-DER	Regional NSGAII	RVEA
C-10/MOP1	1	3	1	10	9	8	6	7	4	5
C-10/MOP2	2	3	1	10	7	9	5	8	4	6
C-10/MOP3	2	1	3	9	8	10	6	4	5	7
C-10/MOP4	1	1	3	6	8	10	7	5	4	9
C-10/MOP5	1	3	1	6	5	10	4	8	6	9
C-10/MOP6	1	2	5	7	5	10	4	8	3	9
C-10/MOP7	3	1	8	1	4	10	6	7	5	9
C-10/MOP8	1	3	2	7	5	10	6	8	4	9
C-10/MOP9	1	3	2	7	5	10	6	8	4	9
IN-1K/MOP1	2	4	1	6	8	9	5	7	3	10
IN-1K/MOP2	2	3	1	6	8	9	4	7	5	10
IN-1K/MOP3	1	1	4	3	8	10	6	7	5	9
IN-1K/MOP4	1	3	1	6	7	9	5	7	4	10
IN-1K/MOP5	1	3	1	6	8	9	5	7	4	10
IN-1K/MOP6	1	3	1	8	6	10	5	7	4	9
IN-1K/MOP7	2	6	1	3	7	10	4	8	5	9
IN-1K/MOP8	1	1	4	1	8	10	5	7	6	9
IN-1K/MOP9	4	1	3	1	8	10	5	6	7	9
Normalized Scores	15.55556	25	23.88889	57.22222	68.88889	96.11111	52.22222	70	45.55556	87.22222
Final Rank	1	3	2	6	7	10	5	8	4	9

Winner Algorithms

CMOSMA_NCHU

A Two Population Evolutionary Framework for Handling NAS Problems

Authors: Chao He, Ming Li, Congxuan Zhang, Hao Chen, Lilin Jie, Leqi Jiang, Junhua Li

Affiliations : Nanchang Hangkong University, Nanjing University of Aeronautics and Astronautics

EABSM-NAS

Evolutionary Algorithm Based on Surrogate Models in Neural Architecture Search

Authors: Chixin Wang, Zhe Wen, Jiajun Chen, Zhen Cui, Boyi Xiao, Weiqin Ying*, Yu Wu*

Affiliations : South China University of Technology, Guangzhou University

DLEA-Niche

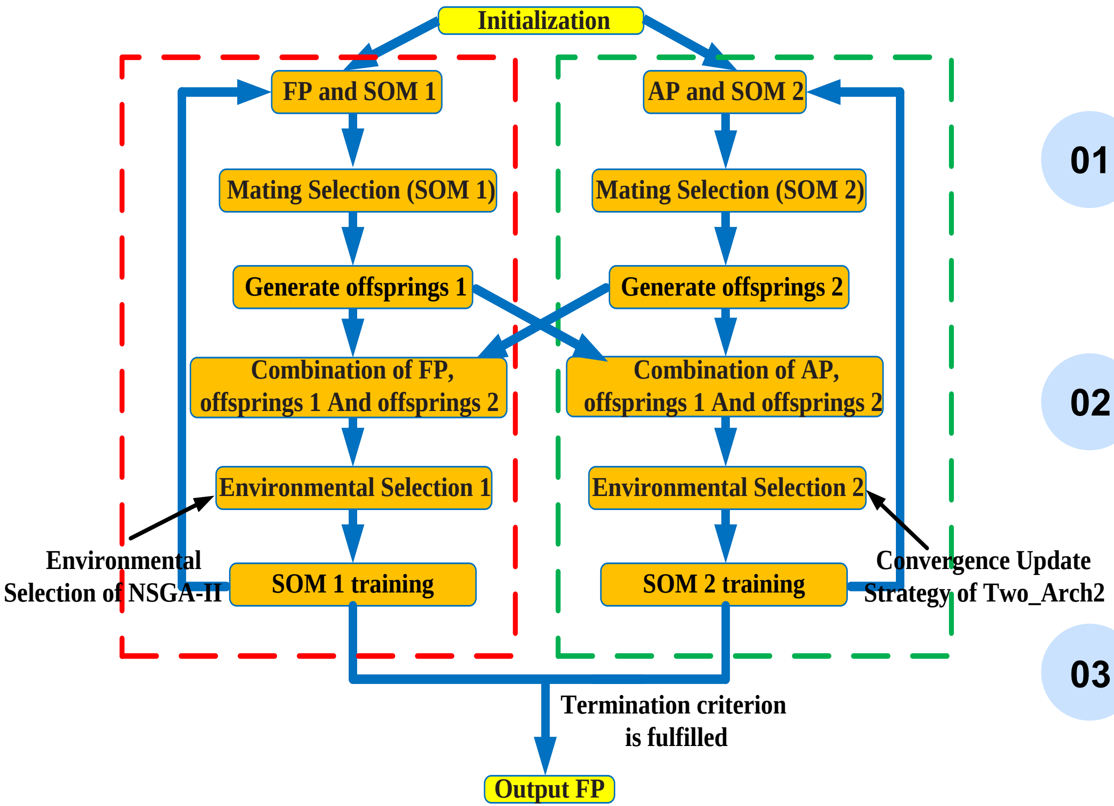
Dynamic Learning Evolutionary Algorithm with Niche-based Diversity Maintenance Strategy

Authors: Gui Li, Guining Zhan

Affiliations : Huazhong University of Science and Technology

CMOSMA_NCHU

A Two Population Evolutionary Framework for Handling NAS Problems[6, 11, 12]



Algorithm Description

01 The unique feature of CMOSMA_NCHU is that Self-Organizing Map (SOM) is embedded into two collaborative and complementary populations during the evolutionary process.

02 Two-SOM collaborative framework is developed to extract neighborhood relationship information of solutions in FP and AP at each generation, respectively. Then, each of the two SOMs is used to guide recombination within the neighborhood to generate promising offspring.

03 FP focuses on promoting population diversity and utilizes the environmental selection strategy of NSGA-II. AP complements FP by exploring unexploited areas. The convergence update strategy of Two_Arch2 is employed as the selection principle for AP to enhance convergence in NAS problems.

EABSM-NAS

Evolutionary Algorithm Based on Surrogate Models in Neural Architecture Search

Probloem Using evolutionary based neural architecture search methods in multiple search spaces.

Search Spaces

NB101 NATS NB201

Tiny: Enumerable

DARTS ResNet50 Transformer MNV3

Large: PB level

Solutions

C-10/MOP1, NATS, NB201

C-10/MOP2

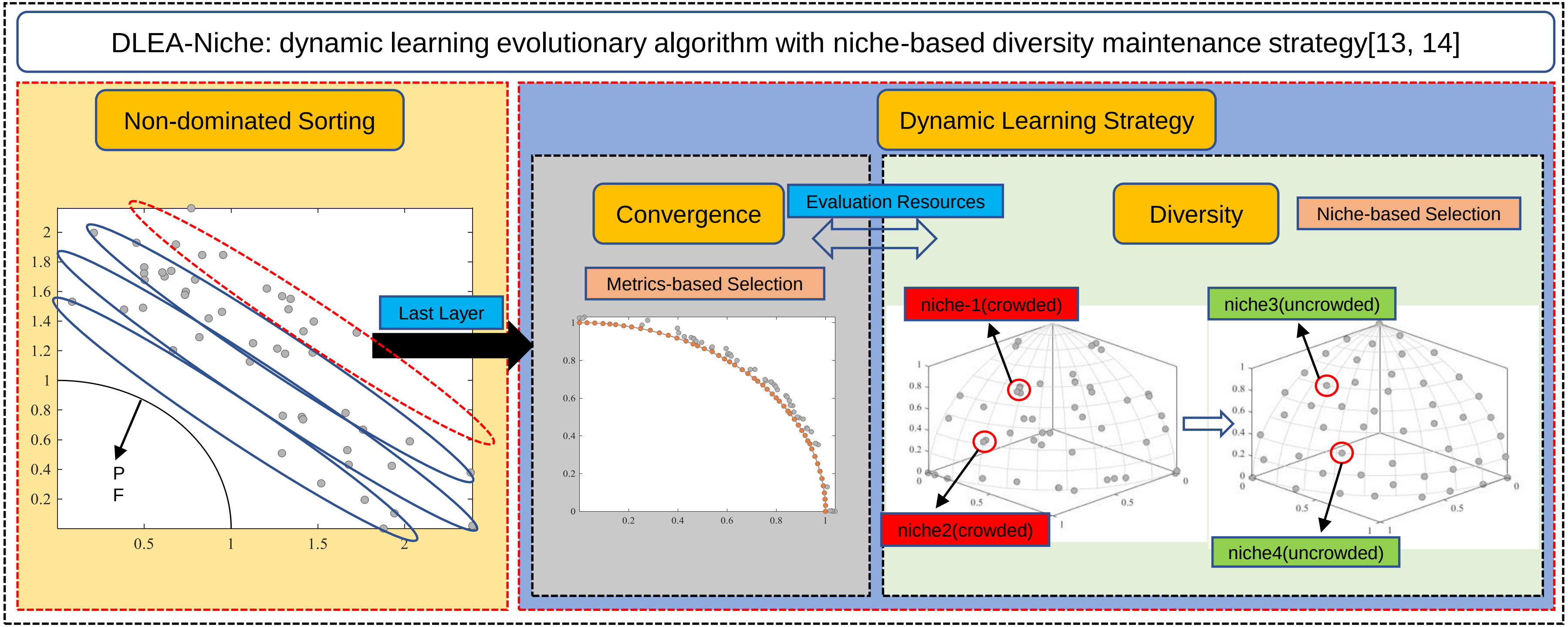
Based on offline surrogate models

1. Use an evolutionary algorithm to evolve and terminate using an early stop strategy.
2. Randomly sample architectures from the search space (or jointly evolve the searched architectures) to train the offline surrogate model.
3. Traverse the entire search space. The other architectures are evaluated using the surrogate model, and a simple sorting strategy is used to rank and get top-k architectures for further evaluation.

Based on online surrogate models

1. Randomly sample sone architectures to train the surrogate model to predict the accuracy, latency and so on.
2. During the evolutionary process, more individuals are generated using the evolutionary algorithm (preset to triple). The trained surrogate model is used to evaluate the architecture and the top-k individuals are taken using the non-dominated sorting.

DLEA-Niche



In each iteration:
Step1: Layer all the solutions that need to be selected using non-dominated sorting;
Step2: For the last non-dominated layer, dynamic learning strategies are used for selection;
Step3: Allocate evaluation resources reasonably according to the current iteration status;
Step4: The convergence-related solutions are selected based on the metrics;
Step5: The diversity-related solutions were selected based on niche-based strategy.

Conclusion

- According to the final ranking based on HV metrics, the winner is CMOSMA_NCHU(A two population evolutionary framework for handling NAS problems)
- The competition showcased the effectiveness of EMO algorithms in addressing multiobjective NAS tasks.
- EMO algorithms offer a promising approach for automated network design in complex application scenarios such as autonomous driving.

Future Extension Directions of EvoXBench

- Enhanced EMO algorithm performance: Investigate novel algorithms that can handle the challenges of NAS.
- More Benchmarking : Consider expanding the range of search spaces, datasets, and hardware configurations.
- Transferability: Explore techniques for transferring knowledge learned from one NAS task to another, reducing the search space and computational costs.

Q & A

Z. Liang (zhenyuliang97@gmail.com) Z. Lu (luzhichaocn@gmail.com)
R. Cheng (ranchengcn@gmail.com) Y. Jin (yaochu.jin@uni-bielefeld.de)

Competition Homepage: https://www.emigroup.tech/index.php/news/ieee-cec2023-competition-on-multiobjective-neural-architecture-search/

Acknowledgements:
This work was supported by the Program for Guangdong Introducing Innovative and Entrepreneurial Teams（Grant No. 2017ZT07X386）.

References

[1] Z. Lu, R. Cheng, Y. Jin, K. C. Tan, and K. Deb, "Neural Architecture Search as Multiobjective Optimization Benchmarks: Problem Formulation and Performance Assessment," *IEEE Transactions on Evolutionary Computation*, 2023.

[2] Y. Tian, R. Cheng, X. Zhang, and Y. Jin, "PlatEMO: A MATLAB platform for evolutionary multi-objective optimization [educational forum]," *IEEE Computational Intelligence Magazine*, vol. 12, no. 4, pp. 73-87, 2017.

[3] J. Bader and E. Zitzler, "HypE: An algorithm for fast hypervolume-based many-objective optimization," *Evolutionary computation*, vol. 19, no. 1, pp. 45-76, 2011.

[4] X. Zhang, Y. Tian, and Y. Jin, "A Knee Point-Driven Evolutionary Algorithm for Many-Objective Optimization," *IEEE Transactions on Evolutionary Computation*, vol. 19, no. 6, pp. 761-776, 2015, doi: 10.1109/tevc.2014.2378512.

[5] Z. Qingfu and L. Hui, "MOEA/D: A Multiobjective Evolutionary Algorithm Based on Decomposition," *IEEE Transactions on Evolutionary Computation*, vol. 11, no. 6, pp. 712-731, 2007, doi: 10.1109/tevc.2007.892759.

[6] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II," *IEEE transactions on evolutionary computation*, vol. 6, no. 2, pp. 182-197, 2002.

[7] R. Cheng, Y. Jin, M. Olhofer, and B. Sendhoff, "A Reference Vector Guided Evolutionary Algorithm for Many-Objective Optimization," *IEEE Transactions on Evolutionary Computation*, vol. 20, no. 5, pp. 773-791, 2016, doi: 10.1109/tevc.2016.2519378.

[8] C. M. Fonseca, L. Paquete, and M. López-Ibánez, "An improved dimension-sweep algorithm for the hypervolume indicator," in *2006 IEEE international conference on evolutionary computation*, 2006: IEEE, pp. 1157-1163.

[9] C. A. Coello Coello and M. Reyes Sierra, "A study of the parallelization of a coevolutionary multi-objective evolutionary algorithm," in *MICAI 2004: Advances in Artificial Intelligence: Third Mexican International Conference on Artificial Intelligence, Mexico City, Mexico, April 26-30, 2004. Proceedings 3*, 2004: Springer, pp. 688-697.

[10] J. Blank and K. Deb, "Pymoo: Multi-objective optimization in python," *IEEE Access*, vol. 8, pp. 89497-89509, 2020.

[11] C. He *et al.*, "A self-organizing map approach for constrained multi-objective optimization problems," *Complex & Intelligent Systems*, vol. 8, no. 6, pp. 5355-5375, 2022.

[12] H. Wang, L. Jiao, and X. Yao, "Two_Arch2: An improved two-archive algorithm for many-objective optimization," *IEEE transactions on evolutionary computation*, vol. 19, no. 4, pp. 524-541, 2014.

[13] M. Li, S. Yang, and X. Liu, "Pareto or non-Pareto: Bi-criterion evolution in multiobjective optimization," *IEEE Transactions on Evolutionary Computation*, vol. 20, no. 5, pp. 645-665, 2015.

[14] G. Li, G.-G. Wang, J. Dong, W.-C. Yeh, and K. Li, "DLEA: A dynamic learning evolution algorithm for many-objective optimization," *Information sciences*, vol. 574, pp. 567-589, 2021.