Robust and Efficient Multi-Fidelity Neural Architecture Search with Zero-Cost Proxy-Guided Local Search (Supplementary Material)

QUAN MINH PHAN and NGOC HOANG LUONG*, University of Information Technology, Viet Nam and Vietnam National University Ho Chi Minh City, Viet Nam

A Local Optima Networks on NAS-Bench-201

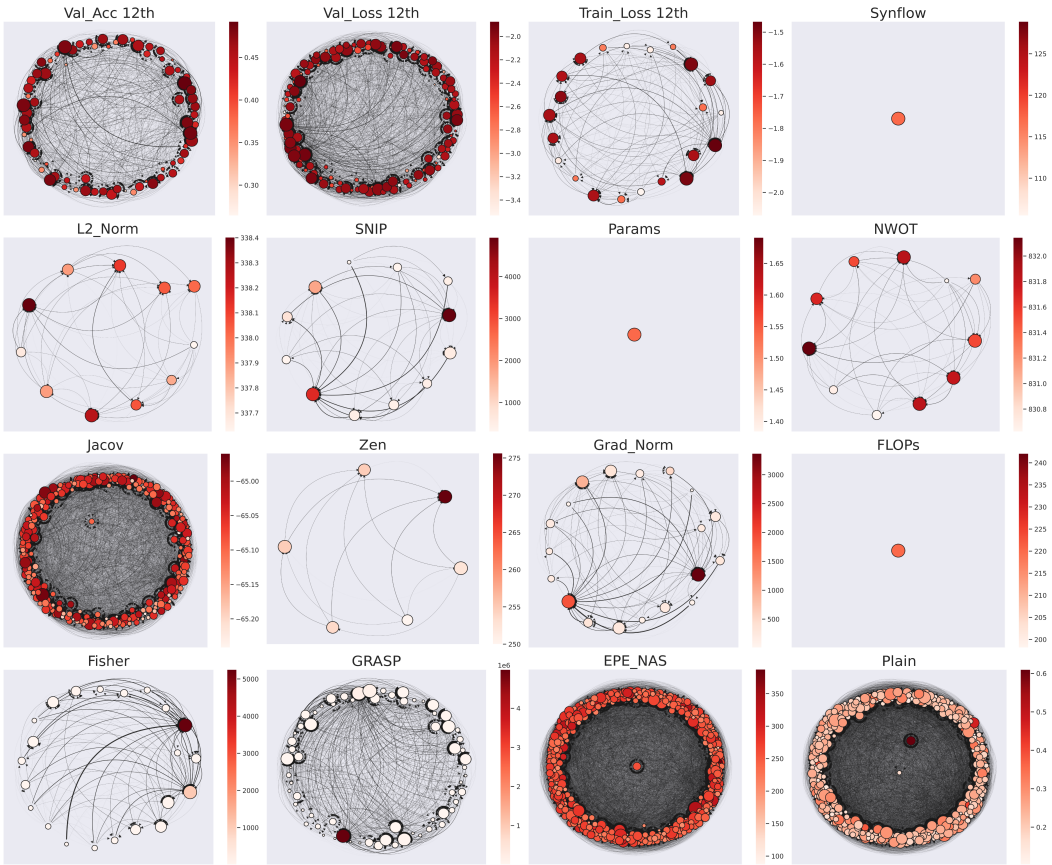

Fig. 1. MLONs of the best-improvement hill climbing algorithm with escape for CIFAR-100 on NAS-Bench-201.

*Corresponding author

Authors' Contact Information: Quan Minh Phan, quanphm@uit.edu.vn; Ngoc Hoang Luong, hoangln@uit.edu.vn, University of Information Technology, Ho Chi Minh City, Viet Nam and Vietnam National University Ho Chi Minh City, Ho Chi Minh City, Viet Nam.

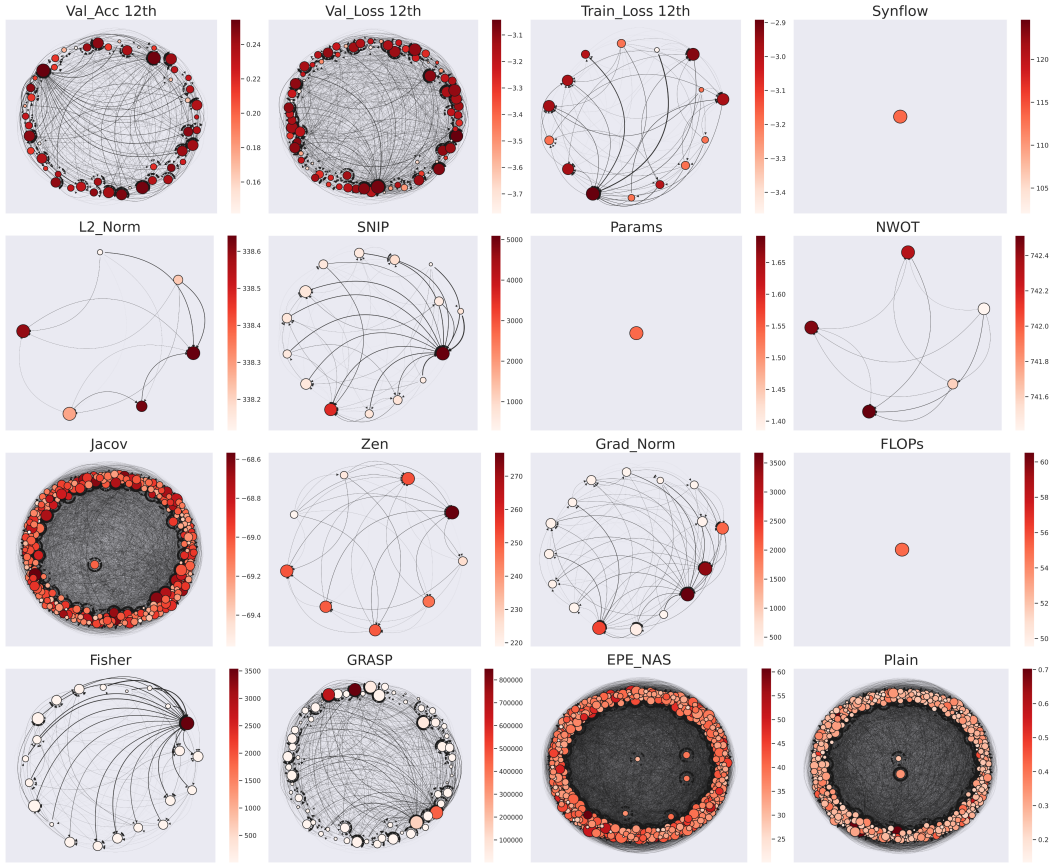

Fig. 2. MLONs of the best-improvement hill climbing algorithm with escape for ImageNet16-120 on NAS-Bench-201.

B Experimental Details

Random Search (RS). In our experiment, the RS iterates the process of randomly sampling the architectures from the search space and evaluating their validation accuracy (or PER for NAS-Bench-ASR) at the 12th epoch till the allowed budget runs out. The architecture that has the highest accuracy so far is the one returned by RS when the search finishes.

Local Search (LS). The local search algorithm implemented in our experiments is the first-improvement hill-climbing algorithm that we employ in the first stage of MF-NAS but use the validation accuracy (or PER for NAS-Bench-ASR) at the 12th epoch as the search objective instead of a ZC proxy. Similar to RS, the procedure of LS is performed till the allowed budget is exhausted and the architecture that has the highest accuracy so far is considered the resulting solution found by LS.

Successive Halving (SH). We refer to [Jamieson and Talwalkar 2016] to implement the SH method in our experiments. For the variant that solely uses SH, the input architectures are randomly sampled from the search space while the input candidates for SH in our MF-NAS framework are obtained via the ZC proxy-guided local search. The number of input architectures k is set to 32 (for NAS-Bench-201) and 16 (for NAS-Bench-101 and NAS-Bench-ASR). For NAS-Bench-201, the architectures are evaluated for their validation accuracies at 5 different epochs: 12, 25, 50, 100, and 200. For NAS-Bench-101 and NAS-Bench-ASR, we evaluate architectures at 4 different epochs: 4, 12, 36, and 108 for NAS-Bench-101, and 10, 20, 30 and 40 for NAS-Bench-ASR. NAS-Bench-201 and NAS-Bench-ASR provide the performance of network architectures at every epoch across 200 and 40 epochs, respectively. As a result, it is possible to perform SH using different sets of epochs beyond the ones we selected for these two benchmarks. In contrast, NAS-Bench-101 only reports the performance of networks at four specific epochs: 4, 12, 36, and 108. Therefore, when executing SH on NAS-Bench-101, we are limited to using only these four checkpoints, as no performance data is available for other epochs.

Regularized Evolution (REA). REA [Real et al. 2019] is an evolution-based algorithm in which individuals in the population represent architectures. In the beginning, REA randomly samples N architectures (in which N is the population size) from the search space to form the initial population. The individual that has the highest fitness (e.g., the highest accuracy (or PER for NAS-Bench-ASR) at the 12th epoch) in the population is selected by using the tournament selection with the tournament size s. The selected individual is then used to produce the new offspring via the mutation operator with probability p_M. The newly-created architecture is added to the population and the oldest individual is eliminated from the population. REA iterates the process of producing and eliminating architectures till the stopping condition is satisfied (e.g., reaching the maximum search time). The individual that has the highest fitness so far during the search is considered as the optimal architecture found by REA. In our experiments, the hyperparameters of REA are set as in [Dong et al. 2022; Dong and Yang 2020], i.e., population size N and tournament size s are 10, the mutation probability p_M is $1/l$ in which l is the length of encoded architecture.

Regularized Evolution + Warmup (REA+W). REA+W uses the Warmup procedure [Abdelfattah et al. 2021] to create the initial population. Specifically, REA+W randomly samples a large number of architectures from the search space and uses the Synaptic Flow metric (Synflow) to evaluate sampled architectures for efficiency. The top-N (in which N is the population size of REA) networks that have the highest Synflow scores are then selected as the individuals of the initial population of REA. The remaining parts of REA+W are executed as in REA. We set the number of sampled architectures in the Warmup stage to 2,000 for all cases.

FreeREA. FreeREA [Cavagnero et al. 2023] is a training-free evolution-based method that combines three zero-cost metrics into a single search objective to guide the Regularized Evolution algorithm. We refer the readers to [Cavagnero et al. 2023] for more details on this method. The hyperparameters of FreeREA are set at their default values in our experiments.

C Performance of MF-NAS with different maximum numbers of training-free evaluations

Table 1. Performance of MF-NAS (Params) with different M-values on NAS problems.

M	NB101-CF10	NB201-CF10	NB201-CF100	NB201-IN16
1,000	93.88 ± 0.25	94.36 ± 0.09	73.51 ± 0.00	46.34 ± 0.06
2,000	93.89 ± 0.25	94.36 ± 0.00	73.51 ± 0.00	46.34 ± 0.00
3,000	93.92 ± 0.22	94.36 ± 0.00	73.51 ± 0.00	46.34 ± 0.00
5,000	93.91 ± 0.20	94.36 ± 0.00	73.51 ± 0.00	46.34 ± 0.00

In this section, we evaluate the performance of the MF-NAS (Params) variants with different maximum numbers of training-free evaluations at the local search stage (i.e., the hyperparameter M), in addition to the default setting of $M = 2{,}000$. The results indicate that although performance varies slightly when M is increased (e.g., $M = 3{,}000$ or $5{,}000$) or decreased (e.g., $M = 1{,}000$), the differences are negligible, even in large-scale search spaces such as NAS-Bench-101 (which contains approximately 423,000 unique networks). Therefore, when faced with a new NAS problem involving a large search space, we recommend using MF-NAS with $M = 2{,}000$, and optionally increasing M to assess whether performance improves further. We note that since all evaluations in the local search stage are training-free, increasing M incurs a negligible computational cost.

D Performance of MF-NAS and R-MF-NAS with Different k Values

Table 2. Test accuracy performance comparison between MF-NAS and R-MF-NAS variants with different k values {32, 48, 64, 96} for the NB201-CF10 problem. The better results are presented in **red**. The results that are significantly worse than the best one (by checking with Student's t-tests at a significance level of 0.01 with Bonferroni correction) are displayed in the gray cells.

Variant	$k = 32$	$k = 48$	$k = 64$	$k = 96$
MF-NAS (**Snip**)	93.13 ± 0.00	93.46 ± 0.09	93.53 ± 0.10	93.56 ± 0.16
R-MF-NAS (**Snip**)	**93.51 ± 0.42**	**93.68 ± 0.32**	**93.73 ± 0.29**	**93.86 ± 0.24**
MF-NAS (**Grasp**)	93.06 ± 0.18	93.31 ± 0.11	93.48 ± 0.16	93.56 ± 0.13
R-MF-NAS (**Grasp**)	**93.54 ± 0.41**	**93.67 ± 0.34**	**93.74 ± 0.30**	**93.86 ± 0.25**
MF-NAS (**Fisher**)	92.21 ± 0.67	93.16 ± 0.13	93.42 ± 0.19	93.49 ± 0.17
R-MF-NAS (**Fisher**)	**93.35 ± 0.45**	**93.53 ± 0.36**	**93.64 ± 0.33**	**93.77 ± 0.28**
MF-NAS (**Plain**)	92.35 ± 0.76	92.54 ± 0.66	92.71 ± 0.51	92.82 ± 0.41
R-MF-NAS (**Plain**)	**92.90 ± 0.68**	**93.12 ± 0.56**	**93.24 ± 0.49**	**93.41 ± 0.42**
MF-NAS (**EPE-NAS**)	**93.73 ± 0.40**	93.78 ± 0.36	**93.89 ± 0.23**	**93.92 ± 0.18**
R-MF-NAS (**EPE-NAS**)	93.63 ± 0.50	**93.80 ± 0.36**	93.86 ± 0.29	**93.93 ± 0.22**
MF-NAS (**Jacov**)	93.61 ± 0.44	93.72 ± 0.30	93.78 ± 0.26	**94.02 ± 0.27**
R-MF-NAS (**Jacov**)	**93.69 ± 0.39**	**93.79 ± 0.30**	**93.86 ± 0.26**	93.93 ± 0.22
MF-NAS (**L2-norm**)	93.45 ± 0.09	93.57 ± 0.03	93.59 ± 0.03	**94.17 ± 0.24**
R-MF-NAS (**L2-norm**)	**93.79 ± 0.35**	**93.90 ± 0.30**	**93.96 ± 0.28**	94.06 ± 0.25
MF-NAS (**Zen**)	89.42 ± 1.04	89.99 ± 0.65	90.26 ± 0.34	90.47 ± 0.50
R-MF-NAS (**Zen**)	**93.99 ± 0.31**	**94.08 ± 0.27**	**94.16 ± 0.22**	**94.24 ± 0.17**
MF-NAS (**Grad-norm**)	93.13 ± 0.06	93.27 ± 0.06	93.32 ± 0.08	93.54 ± 0.10
R-MF-NAS (**Grad-norm**)	**93.47 ± 0.43**	**93.63 ± 0.35**	**93.71 ± 0.33**	**93.85 ± 0.28**
MF-NAS (**NWOT**)	93.58 ± 0.01	93.58 ± 0.01	93.65 ± 0.02	93.66 ± 0.00
R-MF-NAS (**NWOT**)	**93.79 ± 0.38**	**93.91 ± 0.32**	**94.01 ± 0.28**	**94.11 ± 0.24**
MF-NAS (**Synflow**)	**94.36 ± 0.00**	**94.37 ± 0.01**	**94.37 ± 0.02**	**94.37 ± 0.00**
R-MF-NAS (**Synflow**)	94.32 ± 0.08	**94.37 ± 0.01**	**94.37 ± 0.01**	**94.37 ± 0.00**
MF-NAS (**FLOPs**)	**94.36 ± 0.00**	**94.36 ± 0.00**	**94.36 ± 0.00**	**94.37 ± 0.00**
R-MF-NAS (**FLOPs**)	94.05 ± 0.36	94.16 ± 0.31	94.33 ± 0.12	94.36 ± 0.00
MF-NAS (**Params**)	**94.36 ± 0.00**	**94.36 ± 0.00**	**94.36 ± 0.00**	**94.37 ± 0.00**
R-MF-NAS (**Params**)	94.06 ± 0.36	94.25 ± 0.24	94.35 ± 0.07	94.36 ± 0.01
Optimal (in the benchmark)		*94.37*		

Table 3. Test accuracy performance comparison between MF-NAS and R-MF-NAS variants with different k values {32, 48, 64, 96} for the NB201-CF100 problem. The better results are presented in **red**. The results that are significantly worse than the best one (by checking with Student's t-tests at a significance level of 0.01 with Bonferroni correction) are displayed in the gray cells.

Variant	$k = 32$	$k = 48$	$k = 64$	$k = 96$
MF-NAS (**Snip**)	69.25 ± 0.42	69.40 ± 0.21	70.41 ± 0.28	70.89 ± 0.44
R-MF-NAS (**Snip**)	**70.71 ± 1.10**	**71.05 ± 0.92**	**71.32 ± 0.90**	**71.72 ± 0.82**
MF-NAS (**Grasp**)	69.46 ± 0.21	70.12 ± 0.10	70.25 ± 0.16	70.70 ± 0.37
R-MF-NAS (**Grasp**)	**70.79 ± 0.94**	**70.94 ± 0.85**	**71.17 ± 0.80**	**71.44 ± 0.75**
MF-NAS (**Fisher**)	65.60 ± 0.83	69.28 ± 0.28	69.97 ± 0.20	70.26 ± 0.33
R-MF-NAS (**Fisher**)	**70.22 ± 1.07**	**70.57 ± 0.93**	**70.72 ± 0.83**	**71.13 ± 0.74**
MF-NAS (**Plain**)	69.02 ± 1.17	69.40 ± 1.14	69.66 ± 1.05	69.84 ± 0.92
R-MF-NAS (**Plain**)	**69.72 ± 1.19**	**70.08 ± 0.99**	**70.38 ± 0.91**	**70.65 ± 0.86**
MF-NAS (**EPE-NAS**)	**71.44 ± 1.17**	**71.63 ± 0.95**	**71.66 ± 0.90**	71.76 ± 0.73
R-MF-NAS (**EPE-NAS**)	70.86 ± 1.15	71.19 ± 1.01	71.55 ± 0.91	**71.95 ± 0.80**
MF-NAS (**Jacov**)	**71.81 ± 0.55**	71.86 ± 0.53	71.87 ± 0.54	**72.06 ± 0.55**
R-MF-NAS (**Jacov**)	71.67 ± 0.85	**71.92 ± 0.61**	**71.97 ± 0.60**	**72.11 ± 0.58**
MF-NAS (**L2-norm**)	71.28 ± 0.02	71.53 ± 0.07	71.55 ± 0.02	71.56 ± 0.13
R-MF-NAS (**L2-norm**)	**71.67 ± 0.92**	**72.02 ± 0.79**	**72.20 ± 0.79**	**72.44 ± 0.76**
MF-NAS (**Zen**)	60.72 ± 0.75	61.20 ± 0.32	61.20 ± 0.35	61.39 ± 0.94
R-MF-NAS (**Zen**)	**72.23 ± 1.06**	**72.55 ± 0.90**	**72.82 ± 0.76**	**73.13 ± 0.53**
MF-NAS (**Grad-norm**)	68.82 ± 0.81	69.36 ± 0.04	70.15 ± 0.44	70.61 ± 0.39
R-MF-NAS (**Grad-norm**)	**70.66 ± 1.05**	**71.00 ± 0.96**	**71.24 ± 0.87**	**71.58 ± 0.75**
MF-NAS (**NWOT**)	71.51 ± 0.16	71.51 ± 0.19	71.51 ± 0.14	71.53 ± 0.10
R-MF-NAS (**NWOT**)	**71.66 ± 0.88**	**71.88 ± 0.84**	**72.12 ± 0.76**	**72.40 ± 0.69**
MF-NAS (**Synflow**)	**73.51 ± 0.05**	**73.51 ± 0.03**	**73.51 ± 0.02**	**73.51 ± 0.00**
R-MF-NAS (**Synflow**)	**73.51 ± 0.00**	**73.51 ± 0.00**	**73.51 ± 0.00**	**73.51 ± 0.00**
MF-NAS (**FLOPs**)	**73.51 ± 0.00**	**73.51 ± 0.00**	**73.51 ± 0.00**	**73.51 ± 0.00**
R-MF-NAS (**FLOPs**)	72.69 ± 0.92	73.16 ± 0.64	73.47 ± 0.24	**73.51 ± 0.00**
MF-NAS (**Params**)	**73.51 ± 0.00**	**73.51 ± 0.00**	**73.51 ± 0.00**	**73.51 ± 0.00**
R-MF-NAS (**Params**)	72.58 ± 0.97	73.19 ± 0.61	73.46 ± 0.27	**73.51 ± 0.00**
Optimal (in the benchmark)		*73.51*		

Table 4. Test accuracy performance comparison between MF-NAS and R-MF-NAS variants with different k values {32, 48, 64, 96} for the NB201-IN16 problem. The better results are presented in **red**. The results that are significantly worse than the best one (by checking with Student's t-tests at a significance level of 0.01 with Bonferroni correction) are displayed in the gray cells.

Variant	$k = 32$	$k = 48$	$k = 64$	$k = 96$
MF-NAS (**Snip**)	35.96 ± 0.31	40.41 ± 1.66	44.43 ± 0.35	45.40 ± 0.18
R-MF-NAS (**Snip**)	**44.91 ± 1.21**	**45.32 ± 0.97**	**45.60 ± 0.79**	**45.94 ± 0.64**
MF-NAS (**Grasp**)	42.82 ± 1.43	**45.39 ± 0.07**	45.39 ± 0.02	45.49 ± 0.29
R-MF-NAS (**Grasp**)	**44.77 ± 1.30**	45.23 ± 0.96	**45.50 ± 0.83**	**45.86 ± 0.73**
MF-NAS (**Fisher**)	35.97 ± 0.06	41.41 ± 2.76	43.33 ± 0.83	45.24 ± 0.36
R-MF-NAS (**Fisher**)	**44.29 ± 1.52**	**44.88 ± 1.12**	**45.18 ± 0.97**	**45.62 ± 0.79**
MF-NAS (**Plain**)	42.82 ± 2.49	43.65 ± 2.10	43.98 ± 1.76	44.26 ± 1.35
R-MF-NAS (**Plain**)	**43.03 ± 2.07**	**43.89 ± 1.64**	**44.24 ± 1.37**	**44.77 ± 1.06**
MF-NAS (**EPE-NAS**)	44.28 ± 1.22	44.62 ± 0.95	45.00 ± 0.80	**45.35 ± 0.70**
R-MF-NAS (**EPE-NAS**)	**44.32 ± 1.34**	**44.76 ± 1.13**	**45.02 ± 0.99**	45.30 ± 0.85
MF-NAS (**Jacov**)	**44.47 ± 0.98**	**45.23 ± 0.72**	**45.60 ± 0.64**	**45.85 ± 0.49**
R-MF-NAS (**Jacov**)	44.45 ± 1.15	44.79 ± 1.04	45.14 ± 0.95	45.55 ± 0.79
MF-NAS (**L2-norm**)	**46.48 ± 0.21**	45.86 ± 0.40	45.91 ± 0.42	**46.52 ± 0.10**
R-MF-NAS (**L2-norm**)	45.72 ± 0.66	**46.04 ± 0.58**	**46.20 ± 0.53**	46.38 ± 0.42
MF-NAS (**Zen**)	40.77 ± 0.00	40.77 ± 0.00	40.77 ± 0.00	40.80 ± 0.39
R-MF-NAS (**Zen**)	**45.73 ± 0.82**	**46.03 ± 0.62**	**46.16 ± 0.48**	**46.36 ± 0.40**
MF-NAS (**Grad-norm**)	35.97 ± 0.00	40.16 ± 1.01	43.68 ± 1.15	45.32 ± 0.16
R-MF-NAS (**Grad-norm**)	**44.77 ± 1.32**	**45.26 ± 0.98**	**45.52 ± 0.80**	**45.91 ± 0.59**
MF-NAS (**NWOT**)	45.33 ± 0.09	45.93 ± 0.33	**46.18 ± 0.43**	**46.49 ± 0.15**
R-MF-NAS (**NWOT**)	**45.72 ± 0.65**	**46.12 ± 0.60**	46.07 ± 0.58	46.26 ± 0.55
MF-NAS (**Synflow**)	46.34 ± 0.00	46.48 ± 0.02	46.49 ± 0.04	**46.65 ± 0.11**
R-MF-NAS (**Synflow**)	**46.41 ± 0.14**	**46.52 ± 0.11**	**46.55 ± 0.13**	46.59 ± 0.15
MF-NAS (**FLOPs**)	**46.34 ± 0.00**	46.38 ± 0.06	46.44 ± 0.09	**46.59 ± 0.12**
R-MF-NAS (**FLOPs**)	45.86 ± 0.98	**46.30 ± 0.55**	46.47 ± 0.23	46.54 ± 0.20
MF-NAS (**Params**)	**46.34 ± 0.00**	**46.38 ± 0.06**	**46.47 ± 0.11**	**46.62 ± 0.12**
R-MF-NAS (**Params**)	45.82 ± 1.05	46.29 ± 0.56	**46.48 ± 0.24**	46.54 ± 0.20
Optimal (in the benchmark)		*47.31*		

References

Mohamed S. Abdelfattah, Abhinav Mehrotra, Lukasz Dudziak, and Nicholas Donald Lane. 2021. Zero-Cost Proxies for Lightweight NAS. In *9th International Conference on Learning Representations, ICLR 2021*. OpenReview.net.

Niccolò Cavagnero, Luca Robbiano, Barbara Caputo, and Giuseppe Averta. 2023. FreeREA: Training-Free Evolution-based Architecture Search. In *IEEE/CVF Winter Conference on Applications of Computer Vision, WACV 2023*. IEEE, 1493–1502.

Xuanyi Dong, Lu Liu, Katarzyna Musial, and Bogdan Gabrys. 2022. NATS-Bench: Benchmarking NAS Algorithms for Architecture Topology and Size. *IEEE Trans. Pattern Anal. Mach. Intell.* 44, 7 (2022), 3634–3646.

Xuanyi Dong and Yi Yang. 2020. NAS-Bench-201: Extending the Scope of Reproducible Neural Architecture Search. In *International Conference on Learning Representations (ICLR)*.

Kevin G. Jamieson and Ameet Talwalkar. 2016. Non-stochastic Best Arm Identification and Hyperparameter Optimization. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*. 240–248.

Esteban Real, Alok Aggarwal, Yanping Huang, and Quoc V. Le. 2019. Regularized Evolution for Image Classifier Architecture Search. In *AAAI Conference on Artificial Intelligence (AAAI)*. 4780–4789.