GP-NAS-ensemble: a model for the NAS Performance Prediction

Kunlong Chen[1]   Liu Yang[2]   Yitian Chen[3]   Kunjin Chen[4]   Yidan Xu[1]   Lujun Li[5]

[1]Meituan   [2]Tencent   [3]Bigo Technology   [4]Alibaba Group   [5]Chinese Academy of Sciences

[1]{chenkunlong, xuyidan02}@meituan.com, [2] arielwillow@163.com,

[3]yitiansky@gmail.com [4]kunjin.ckj@alibaba-inc.com, [6]lilujunai@gmail.com

Abstract

It is of great significance to estimate the performance of a given model architecture without training in the application of Neural Architecture Search (NAS) as it may take a lot of time to evaluate the performance of an architecture. In this paper, a novel NAS framework called GP-NAS-ensemble is proposed to predict the performance of a neural network architecture with a small training dataset. We make several improvements on the GP-NAS model to make it share the advantage of ensemble learning methods. Our method ranks the 2nd place in the Performance Prediction Track of CVPR2022 Second lightweight NAS challenge.

1. Introduction

With the development of Neural Architecture Search (NAS) techniques, it becomes popular to design deep neural network architectures automatically [1]. It is a critical part to design a performance predictor to analyze the relationship between model architecture and its accuracy evaluated on certain tasks for NAS algorithms, as training a large net can be fairly expensive [42].

GP-NAS-ensemble model is proposed as a novel predictor in this paper. Based on GP-NAS model [16], we make several improvements to make it more accurate and robust. The validity of our method is verified in the Performance Prediction Track of CVPR2022 Second lightweight NAS challenge.

1.1. Literature Review

General NAS. In a wide range of computer vision tasks [3, 11, 22, 38, 40], manually constructed neural networks have had great success. Artificial designs are usually thought to be suboptimal. Both academia and industry have recently been more interested in neural architecture search (NAS). Early efforts made use of reinforcement learning [30, 43–45] and evolutionary algorithms [19, 26, 27, 29, 37] and discovered several high-cost and high-performance designs. Later work aims to lower the cost of searching while

increasing performance, which can be divided into three categories: one-shot NAS [6, 8, 12, 41], gradient-based approaches [5, 18, 33], and predictor NAS, which differ in the network architecture modeling process. One-Shot NAS [6] firstly trains an over-parameterized supernet, then searches a discrete search space that includes numerous candidate models. The sample strategy is important during the training stage, since it determines how to train an effective supernet for performance estimation. Gradient-based methods incorporate the architecture parameters for each operator and use backpropagation to jointly optimize them and the weights of the network.

Predictor NAS. Predictor NAS approaches attempt to accurately and effectively forecast the performance of a particular neural network. These procedures learns the accuracy predictor by sampling pairs of architectures and their accuracies. Some works [4, 7, 13, 14] extend this line of thought by training a predictor to extrapolate the NAS learning curves. The regression problem [35] or ranking [23, 39] can be used to describe the goal of fitting the predictor [10, 20, 21, 34]. Moreover, feature representation and sampling methods are crucial for search performance. [21] uses acyclic graphs in a continuous space of potential embeddings along with performance predictors. [32] improves the two-stage NAS with Pareto-aware sampling strategies. [28] uses Bayesian regression as a proxy model to select candidates, and [36] replaces a strong predictor with a set of weaker predictors.

Multi-task NAS. Multi-task learning refers to different tasks sharing part of the network backbone or weights. These tasks are often able to learn from each other to achieve better performance and training efficiency. Recent neural network architecture search methods and benchmarks [9, 13, 31] for multi-task and cross-task have attracted a lot of attention from the community. Despite being underappreciated in comparison to single-task NAS, there are still several excellent algorithms. [24] uses continuous learning to find a single cell structure that can generalize well to unknown tasks via multi-task architecture search based on the weight sharing technique. [17] used gradient-based NAS to

find the best cell structure for a variety of autonomous driving tasks. [15] proposed to construct graphs from datasets in a meta-learning approach to make the methods generalize well across numerous tasks.

2. Proposed Method

In this paper, we introduce the *GP-NAS-ensemble* model [16], which ranked 2nd in the CVPR 2022 NAS competition: performance estimation track. Based on the GP-NAS model, we aim at establishing a model with better performance using ensemble learning technique.

2.1. GP-NAS

GP-NAS is a powerful method to predict the performance of a neural network given its architecture, especially when the size of training data is small [16]. To be more specific, the GP-NAS model uses the Gaussian process regression model to predict the accuracy of a neural network model under the assumption that the joint distribution between the training observations $\mathbf{y}$ and the test function values $f(\boldsymbol{X}_*) = \boldsymbol{f}_*$ is [25]

$$p\left(\boldsymbol{y}, \boldsymbol{f}_* \mid \boldsymbol{X}, \boldsymbol{X}_*\right) = \mathcal{N}\left(\left[\begin{array}{c} m(\boldsymbol{X}) \\ m(\boldsymbol{X}_*) \end{array}\right], \left[\begin{array}{cc} \mathbf{K} + \sigma_n^2 \mathbf{I} & k\left(\boldsymbol{X}', \boldsymbol{X}_*\right) \\ k\left(\boldsymbol{X}_*, \boldsymbol{X}\right) & k\left(\boldsymbol{X}_*, \boldsymbol{X}_*\right). \end{array}\right]\right) \tag{1}$$

We then have

$$p\left(\boldsymbol{f}_* \mid \boldsymbol{X}, \boldsymbol{y}, \boldsymbol{X}_*\right) = \mathcal{N}\left(\mathbb{E}\left[\boldsymbol{f}_* \mid \boldsymbol{X}, \boldsymbol{y}, \boldsymbol{X}_*\right], \mathbb{V}\left[\boldsymbol{f}_* \mid \boldsymbol{X}, \boldsymbol{y}, \boldsymbol{X}_*\right]\right),$$
$$\mathbb{E}\left[\boldsymbol{f}_* \mid \boldsymbol{X}, \boldsymbol{y}, \boldsymbol{X}_*\right] = m_{\text{post}}\left(\boldsymbol{X}_*\right) = \underbrace{m\left(\boldsymbol{X}_*\right)}_{\text{prior mean}} + \underbrace{k\left(\boldsymbol{X}_*, \boldsymbol{X}\right)\left(\boldsymbol{K} + \sigma_n^2 \boldsymbol{I}\right)^{-1}}_{\text{"Kalman gain"}} \underbrace{(\boldsymbol{y} - m(\boldsymbol{X}))}_{\text{error}}, \tag{2}$$

where $\mathbf{K}$ is the Gram matrix, which contains the kernel functions evaluating on all pairs of data points, σ_n^2 is the noise variance of observations.

Simply put, the Gaussian process model uses the prior mean as an initial guess for the target value. The guess is then corrected by a mechanism similar to Kalman filter. It is therefore important to note that, the cores of the GP-NAS method are two parts: (1) the estimation of the prior mean; (2) the specific formula of the kernel function. In the original implementation of the algorithm, the linear regression method is used to estimate the prior mean with a variant of the radial basis function (RBF) kernel:

$$m(\mathbf{x}) = \mathbf{w} \cdot \mathbf{x},$$
$$k_{rbf}^s(\mathbf{x}_1, \mathbf{x}_2) = \exp[-\sqrt{||(\mathbf{x}_1 - \mathbf{x}_2)||}/l], \tag{3}$$

where $\mathbf{w}$ is the coefficient vector of the linear regression model and l is the length scale of the kernel.

2.2. Feature Engineering

In the original dataset, an ordinal encoder is used to represent each architecture. For example, the search space of features such as depth of network, number of heads of each layer, and MLP ratio of each layer are $\{10, 11, 12\}$. They are encoded as $\{1, 2, 3\}$, respectively, in the feature space.

One-hot encoding is a sparse way of representing data in a binary string in which only a single bit can be 1, while

original feature	one-hot encoding	two-hot encoding
0	$[0, 0, 0, 0]$	$[0, 0, 0, 0]$
1	$[1, 0, 0, 0]$	$[1, 1, 0, 0]$
2	$[0, 1, 0, 0]$	$[0, 1, 1, 0]$
3	$[0, 0, 1, 0]$	$[0, 0, 1, 1]$

Table 1. The feature engineering methods used in our framework.

all others are 0 [2]. This method is considered as a popular way to deal with catogorical features in the machine learning community. However, the disadvantage of one-hot encoding is that it does not contain the information of the 'similarity' of two data points. For instance, the distance between "3" and "1" should be larger than the distance between "2" and "1". In order to solve this problem, we also use the two-hot encoding method to represent the similarity in the feature space. An example of applying one-hot encoding and two-hot encoding methods is shown in Table 1.

2.3. Label transformation

The dataset does not provide us with the prediction accuracy of each architecture on test dataset. Instead, we only have access to the relative rank of each data point. We believe that it is more natural to predict the accuracy of each model architecture instead of predicting ranks through several trials. Since we do not have enough domain knowledge
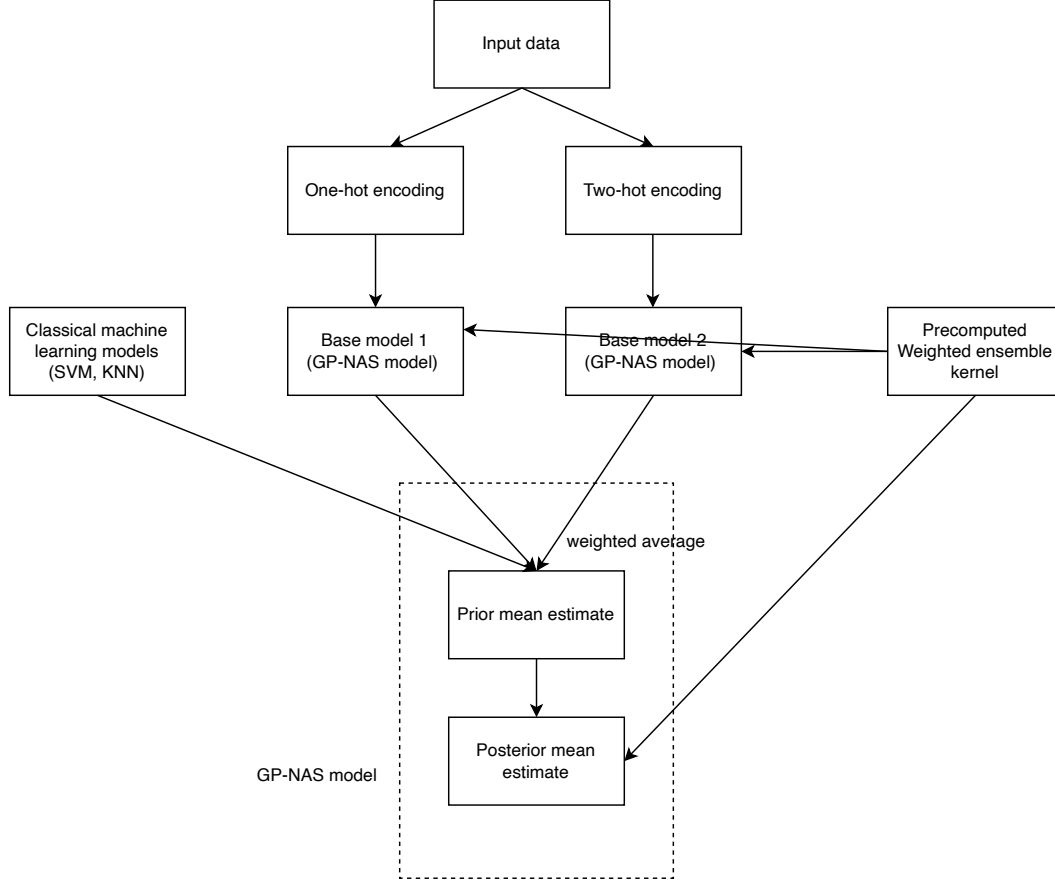
Fig. 1. An illustration of the proposed GP-NAS-ensemble model.

Task	kernel length	weighted kernel ratio	Label transformation method	set of base learner
task 0	22	(0.18, 0.82)	Normal distribution	GP-NAS, KNN
task 1	28	(0.62, 0.38)	Left-skewed Normal distribution	GP-NAS, SVR
task 2	24	(0.02, 0.98)	Left-skewed Normal distribution	GP-NAS, SVR
task 3	25	(0.6, 0.4)	Normal distribution	GP-NAS, SVR
task 4	22	(0.7, 0.3)	Left-skewed Normal distribution	GP-NAS, SVR
task 5	22	(0.3, 0.7)	Normal distribution	GP-NAS, SVR
task 6	22	-	Normal distribution	GP-NAS
task 7	22	(0.3, 0.7)	Normal distribution	GP-NAS, SVR

Table 2. The configurations for each task.

on this field, we need to guess the probability distribution of model performances and then assign each model with a score by sampling from the proposed distribution. There are two ways to guess the distribution of the accuracy: (1) they follow a normal distribution, (2) they follow a left-skewed bell-shaped distribution. We decide the distribution by conducting trials on the public leaderboard.

2.4. Weighted ensemble kernel function

The idea behind the weighted kernel function is that in each task, we should focus on different parts of the feature set. The weighted kernel function is

$$k_w(\mathbf{x}_1, \mathbf{x}_2) = \exp[-\sqrt{(\mathbf{x}_1 - \mathbf{x}_2)^T I_w (\mathbf{x}_1 - \mathbf{x}_2)}/l], \quad (4)$$

where I_w is a diagonal matrix. In order to estimate I_w for each task, we use the Bayesian optimization method to max-

Modifications	score (public)
GP-NAS	0.668
+ feature engineering	0.787
+ label transformation	0.796
+ ensemble learning	0.798
+ weighted ensemble kernel	0.800

Table 3. The model performance on the public leaderboard.

imize the Kendall rank correlation coefficient on the training dataset. Considering the fact that only a small amount of data is available, it may be prone to over-fitting by applying Bayesian optimization method directly. We know that if k_1, k_2 are valid kernels, $k_1 + k_2$ is still a valid kernel [25]. We thus proposed a new kernel k_w^e, which is the weighted sum of k_w and k_{rbf}^s:

$$k_w^e(\mathbf{x}_1, \mathbf{x}_2) = \beta_1 k_{rbf}^s(\mathbf{x}_1, \mathbf{x}_2) + \beta_2 k_w(\mathbf{x}_1, \mathbf{x}_2), \quad (5)$$

where the values of β_1 and β_2 are selected by their performances on the public leaderboard.

2.5. GP-NAS-ensemble model

In this section, we briefly introduce the structure of our proposed GP-NAS-ensemble model. It adopts the modifications described in previous sections. The basic scheme of the GP-NAS-ensemble model is shown in Figure 1, which contains the following steps:

1. Weighted ensemble kernel computing. We use the Bayesian optimization method to estimate the most suitable weighted ensemble kernel function for each task.

2. Base model establishment. Two GP-NAS models are built as base models. The difference lies in that we feed the data with one-hot encoding to the first model and we feed the data with two-hot encoding to the second one. In addition, several classical supervised learning methods are also considered as parts of our base-model set since we need to enhance the diversity between base models.

3. Base model training. We train each base model separately on the training dataset.

4. Ensemble model establishment. A GP-NAS-ensemble model can be built on top of the base models mentioned above. To be more specific, we predict the prior mean of a given test data by averaging the output of each base model. The estimation of the posterior mean is the same as the basic GP-NAS model.

3. Experiments

The configurations of our final submission except for task 6 are shown in Table 2. Specifically, we observe that the above-mentioned weighted ensemble kernel doesn't not increase the score of task 6. Therefore, we use nine-hot encoding method for the feature engineering step of task 6. The configurations are tuned based on scores on the public leaderboard.

In Table 3, we show the results of an ablation study which compares the proposed model with the original GP-NAS model. The GP-NAS model can achieve about 0.668 on public leaderboard. If we transform input features with one-hot encoding or two-hot encoding, the score is increased to about 0.787. After label transformation with normal distribution, we obtain a score about 0.796. When all modifications are applied, we get a final score close to 0.800.

4. Conclusion

Unlike most other competitors of this competition who use deep-learning method or other classical supervised machine learning to achieve a high score, we fully explored the potential of GP-NAS model with only small modifications on the model architecture and the feature engineering pipeline. The score of the proposed method on the public leaderboard increased from 0.668 to about 0.800.

References

[1] https://aistudio.baidu.com/aistudio/competition/detail/150/0/introduction. 1

[2] https://hackernoon.com/what-is-one-hot-encoding-why-and-when-do-you-have-to-use-it-e3c6186d008f. 2

[3] Pattathal V. Arun, Ittai Herrmann, Krishna M. Budhiraju, and Arnon Karnieli. Convolutional network architectures for super-resolution/sub-pixel mapping of drone-derived images. *Pattern Recognition*, 88:431–446, 2019. 1

[4] Bowen Baker, Otkrist Gupta, Ramesh Raskar, and Nikhil Naik. Accelerating neural architecture search using performance prediction. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Workshop Track Proceedings*. OpenReview.net, 2018. 1

[5] Jianlong Chang, Yiwen Guo, MENG Gaofeng, Zhouchen Lin, Shiming XIANG, Chunhong Pan, et al. Data: Differentiable architecture approximation with distribution guided sampling. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2020. 1

[6] Xiangxiang Chu, Bo Zhang, Ruijun Xu, and Jixiang Li. Fairnas: Rethinking evaluation fairness of weight sharing neural architecture search. *arXiv preprint arXiv:1907.01845*, 2019. 1

[7] Tobias Domhan, Jost Tobias Springenberg, and Frank Hutter. Speeding up automatic hyperparameter optimization of

deep neural networks by extrapolation of learning curves. In *Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence, IJCAI 2015, Buenos Aires, Argentina, July 25-31, 2015*, pages 3460–3468. AAAI Press, 2015. 1

[8] Xuanyi Dong and Yi Yang. One-shot neural architecture search via self-evaluated template network. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 3681–3690, 2019. 1

[9] Yawen Duan, Xin Chen, Hang Xu, Zewei Chen, Xiaodan Liang, Tong Zhang, and Zhenguo Li. Transnas-bench-101: Improving transferability and generalizability of cross-task neural architecture search. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2021, virtual, June 19-25, 2021*, pages 5251–5260. Computer Vision Foundation / IEEE, 2021. 1

[10] Lukasz Dudziak, Thomas C. P. Chau, Mohamed S. Abdelfattah, Royson Lee, Hyeji Kim, and Nicholas D. Lane. BRP-NAS: prediction-based NAS using gcns. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020. 1

[11] Jiuxiang Gu, Zhenhua Wang, Jason Kuen, Lianyang Ma, Amir Shahroudy, Bing Shuai, Ting Liu, Xingxing Wang, Gang Wang, Jianfei Cai, and Tsuhan Chen. Recent advances in convolutional neural networks. *Pattern Recognition*, 77:354–377, 2018. 1

[12] Zichao Guo, Xiangyu Zhang, Haoyuan Mu, Wen Heng, Zechun Liu, Yichen Wei, and Jian Sun. Single path one-shot neural architecture search with uniform sampling. In *Proceedings of the European Conference on Computer Vision*, pages 818–833, 2020. 1

[13] Minbin Huang, Zhijian Huang, Changlin Li, Xin Chen, Hang Xu, Zhenguo Li, and Xiaodan Liang. Arch-graph: Acyclic architecture relation predictor for task-transferable neural architecture search. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2022. 1

[14] Aaron Klein, Stefan Falkner, Jost Tobias Springenberg, and Frank Hutter. Learning curve prediction with bayesian neural networks. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net, 2017. 1

[15] Hayeon Lee, Eunyoung Hyung, and Sung Ju Hwang. Rapid neural architecture search by learning to generate graphs from datasets. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, 2021. 2

[16] Zhihang Li, Teng Xi, Jiankang Deng, Gang Zhang, Shengzhao Wen, and Ran He. GP-NAS: Gaussian process based neural architecture search. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11933–11942, 2020. 1, 2

[17] Hao Liu, Dong Li, Jinzhang Peng, Qingjie Zhao, Lu Tian, and Yi Shan. MTNAS: search multi-task networks for autonomous driving. In *Computer Vision - ACCV 2020 - 15th Asian Conference on Computer Vision, Kyoto, Japan, November 30 - December 4, 2020, Revised Selected Papers,*

Part III, volume 12624 of *Lecture Notes in Computer Science*, pages 670–687. Springer, 2020. 1

[18] Hanxiao Liu, Karen Simonyan, and Yiming Yang. Darts: Differentiable architecture search. In *International Conference on Learning Representations*, 2019. 1

[19] Zhichao Lu, Ian Whalen, Vishnu Boddeti, Yashesh D. Dhebar, Kalyanmoy Deb, Erik D. Goodman, and Wolfgang Banzhaf. NSGA-NET: A multi-objective genetic algorithm for neural architecture search. *CoRR*, abs/1810.03522, 2018. 1

[20] Renqian Luo, Xu Tan, Rui Wang, Tao Qin, Enhong Chen, and Tie-Yan Liu. Neural architecture search with GBDT. *CoRR*, abs/2007.04785, 2020. 1

[21] Renqian Luo, Fei Tian, Tao Qin, Enhong Chen, and Tie-Yan Liu. Neural architecture optimization. In *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*, pages 7827–7838, 2018. 1

[22] Ningning Ma, Xiangyu Zhang, Hai-Tao Zheng, and Jian Sun. Shufflenet v2: Practical guidelines for efficient cnn architecture design. In *Proceedings of the European Conference on Computer Vision*, pages 116–131, 2018. 1

[23] Xuefei Ning, Yin Zheng, Tianchen Zhao, Yu Wang, and Huazhong Yang. A generic graph-based neural architecture encoding scheme for predictor-based NAS. In *Computer Vision - ECCV 2020 - 16th European Conference, Glasgow, UK, August 23-28, 2020, Proceedings, Part XIII*, volume 12358 of *Lecture Notes in Computer Science*, pages 189–204. Springer, 2020. 1

[24] Ramakanth Pasunuru and Mohit Bansal. Continual and multi-task architecture search. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 1911–1922. Association for Computational Linguistics, 2019. 1

[25] Carl Edward Rasmussen. Gaussian processes in machine learning. In *Summer school on machine learning*, pages 63–71. Springer, 2003. 2, 4

[26] Esteban Real, Alok Aggarwal, Yanping Huang, and Quoc V. Le. Regularized evolution for image classifier architecture search. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019*, pages 4780–4789. AAAI Press, 2019. 1

[27] Esteban Real, Sherry Moore, Andrew Selle, Saurabh Saxena, Yutaka Leon Suematsu, Jie Tan, Quoc V. Le, and Alexey Kurakin. Large-scale evolution of image classifiers. In *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, volume 70 of *Proceedings of Machine Learning Research*, pages 2902–2911. PMLR, 2017. 1

[28] Han Shi, Renjie Pi, Hang Xu, Zhenguo Li, James T. Kwok, and Tong Zhang. Bridging the gap between sample-based

and one-shot neural architecture search with BONAS. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020. 1

[29] Masanori Suganuma, Mete Ozay, and Takayuki Okatani. Exploiting the potential of standard convolutional autoencoders for image restoration by evolutionary search. In *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, volume 80 of *Proceedings of Machine Learning Research*, pages 4778–4787. PMLR, 2018. 1

[30] Mingxing Tan, Bo Chen, Ruoming Pang, Vijay Vasudevan, Mark Sandler, Andrew Howard, and Quoc V. Le. Mnasnet: Platform-aware neural architecture search for mobile. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019*, pages 2820–2828. Computer Vision Foundation / IEEE, 2019. 1

[31] Renbo Tu, Mikhail Khodak, Nicholas Roberts, and Ameet Talwalkar. Nas-bench-360: Benchmarking diverse tasks for neural architecture search. *CoRR*, abs/2110.05668, 2021. 1

[32] Dilin Wang, Meng Li, Chengyue Gong, and Vikas Chandra. Attentivenas: Improving neural architecture search via attentive sampling. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2021, virtual, June 19-25, 2021*, pages 6418–6427. Computer Vision Foundation / IEEE, 2021. 1

[33] Naiyan Wang, Shiming XIANG, Chunhong Pan, et al. You only search once: Single shot neural architecture search via direct sparse optimization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2020. 1

[34] Chen Wei, Chuang Niu, Yiping Tang, and Jimin Liang. NPE-NAS: neural predictor guided evolution for neural architecture search. *CoRR*, abs/2003.12857, 2020. 1

[35] Wei Wen, Hanxiao Liu, Yiran Chen, Hai Helen Li, Gabriel Bender, and Pieter-Jan Kindermans. Neural predictor for neural architecture search. In *Computer Vision - ECCV 2020 - 16th European Conference, Glasgow, UK, August 23-28, 2020, Proceedings, Part XXIX*, volume 12374 of *Lecture Notes in Computer Science*, pages 660–676. Springer, 2020. 1

[36] Junru Wu, Xiyang Dai, Dongdong Chen, Yinpeng Chen, Mengchen Liu, Ye Yu, Zhangyang Wang, Zicheng Liu, Mei Chen, and Lu Yuan. Stronger nas with weaker predictors. *arXiv preprint arXiv:2102.10490*, 2021. 1

[37] Lingxi Xie and Alan L. Yuille. Genetic CNN. In *IEEE International Conference on Computer Vision, ICCV 2017, Venice, Italy, October 22-29, 2017*, pages 1388–1397. IEEE Computer Society, 2017. 1

[38] Ting-Bing Xu, Peipei Yang, Xu-Yao Zhang, and Cheng-Lin Liu. Lightweightnet: Toward fast and lightweight convolutional neural networks via architecture distillation. *Pattern Recognition*, 88:272–284, 2019. 1

[39] Yixing Xu, Yunhe Wang, Kai Han, Yehui Tang, Shangling Jui, Chunjing Xu, and Chang Xu. Renas: Relativistic evaluation of neural architecture search. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2021, vir-tual, June 19-25, 2021*, pages 4411–4420. Computer Vision Foundation / IEEE, 2021. 1

[40] Mohamed Yousef, Khaled F Hussain, and Usama S Mohammed. Accurate, data-efficient, unconstrained text recognition with convolutional neural networks. *Pattern Recognition*, 108:107482, 2020. 1

[41] Miao Zhang, Huiqi Li, Shirui Pan, Xiaojun Chang, Chuan Zhou, Zongyuan Ge, and Steven W Su. One-shot neural architecture search: Maximising diversity to overcome catastrophic forgetting. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2020. 1

[42] Ruyi Zhang, Ziwei Yang, Zhi Yang, Xubo Yang, Lei Wang, and Zheyang Li. Cascade bagging for accuracy prediction with few training samples. *arXiv preprint arXiv:2108.05613*, 2021. 1

[43] Zhao Zhong, Junjie Yan, Wei Wu, Jing Shao, and Cheng-Lin Liu. Practical block-wise neural network architecture generation. In *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*, pages 2423–2432. Computer Vision Foundation / IEEE Computer Society, 2018. 1

[44] Barret Zoph and Quoc V. Le. Neural architecture search with reinforcement learning. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net, 2017. 1

[45] Barret Zoph, Vijay Vasudevan, Jonathon Shlens, and Quoc V. Le. Learning transferable architectures for scalable image recognition. In *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*, pages 8697–8710. Computer Vision Foundation / IEEE Computer Society, 2018. 1