2022 IEEE 12th International Conference on RFID Technology and Applications (RFID-TA) | 978-1-6654-6594-6/22/$31.00 ©2022 IEEE | DOI: 10.1109/RFID-TA54958.2022.9924146

Balanced neural architecture search and optimization for specific emitter identification

1st Mingyang Du
College of Electronic Engineering
National University of Defense Technology
Hefei, China
dumingyang17@nudt.edu.cn

2nd Ping Zhong
National Key Laboratory of Science and Technology on ATR
National University of Defense Technology
Changsha, China
zhongping@nudt.edu.cn

3rd Xiaohao Cai
School of Electronics and Computer Science
University of Southampton
Southampton, UK
X.Cai@soton.ac.uk

4th Daping Bi
College of Electronic Engineering
National University of Defense Technology
Hefei, China
bdpeei@163.com

5th Zhifei Li
School of Space Information
Space Engineering University
Beijing, China
lee@seu.email.cn

***Abstract*—Fixed-structure neural network lacks flexibility when tackling different classification tasks, prompting a growing interest in developing automated neural architecture search (NAS) methods. Approaches so far mainly consider the classification accuracy of the searching results for NAS, yet another important factor, the computation cost, is ignored. In this paper, a feasibility problem is modeled subject to specific constraints in terms of both the classification accuracy and computation cost, which can greatly enhance the flexibility against the fixed *"balanced function"* proposed in recent work in identifying radar signals in different electromagnetic environments. Moreover, to be able to traverse the infinite feasible region formed by the constraints, we propose a simple yet effective method based on the Gaussian process regression model by fine-tuning an initialized balanced function and leveraging a data distribution that meets the constraints. Experimental results demonstrate the superiority of the proposed NAS technique in designing comparably accurate network structures against manually-designed models, with less computation cost compared to conventional NAS algorithms.**

***Index Terms*—Specific emitter identification, time-frequency distribution, neural architecture search, Gaussian process**

I. Introduction

Specific emitter identification (SEI) is the procedure of extracting fingerprint features from the received signals to discriminate and classify different emitters, which has been extensively utilized in military communication, radar system, cognitive radio and self-organized network [1]. However, with the fast development of the sensor design theory and manufacturing technique, the high-density radio signals from various emitters make the spatial spectrum more complex, resulting in new challenges for SEI [2]. In other words, SEI in the complex electromagnetic environment is inherently an urgent and extremely challenging problem.

Deep learning techniques, represented by convolutional neural network (CNN) and recurrent neural network (RNN), are proved to be efficient and effective in various tasks such as image and video classification, natural language processing, etc. Meanwhile, it also provides new opportunities for SEI. A typical pathway has drawn a surge of interest in recent years [3], i.e., firstly transferring the time-series radar signals into time-frequency distribution images by time-frequency analysis methods, e.g., short-time Fourier transform (STFT) and Choi-Williams distributions, and then utilizing state-of-the-art (SOTA) neural networks, like Inception-v3 or ResNet-152, to classify emitters via the aforementioned transformation results. Auxiliary feature extraction for the raw radar signal like time-frequency analysis in SEI could reduce the corruption by noise and achieve more robust performance. However, a single network could not handle well multiple data sets with variable signal-to-noise ratio (SNR) in SEI [4]. For example, it is obvious that classifying the signal data with higher SNR will be much easier than that with lower SNR since the random noise existence corrupts the true signal feature. A much deeper network might work well for signal data with intense noise, whereas it might experience performance degradation caused by severe overfitting when tackling clean signal data [5]. It would be ideal if the most suitable classifier could be selected automatically to match the current electromagnetic environment; for example, deeper models are utilized to classify lower SNR signals, while simpler models are adopted to identify signals with higher SNR. Hence, it is urgent to design a mechanism which is capable of automatically constructing optimized network structures to cope with different environments.

Neural architecture search (NAS) methods aiming at automating the machine learning pipeline have been an explosion of research interest in recent years [6]. Some SOTA algorithms, e.g. [7], could design models achieving comparable performance as manual artworks; however, most of them are extremely time consuming and computationally expensive; e.g., the method presented in [8] takes 3150 GPU days and 450 GPUs for CIFAR-10 and ImageNet in the searching procedure. Following the fundamental NAS structure in [9], a *balanced function* was presented in [10] which takes both the validation accuracy and computation budget into consideration while

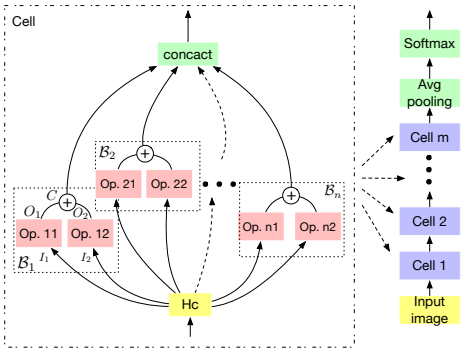


Fig. 1. Blocks and cells in a full CNN. "Op" is the abbreviation of operation, which includes convolution, pooling and identity mapping, with different kernel sizes. "Hc" denotes the output of the previous hidden layer in a cell. $\mathcal{B}_1, \mathcal{B}_2, ..., \mathcal{B}_n$ are individual blocks, and together form the cell structure.

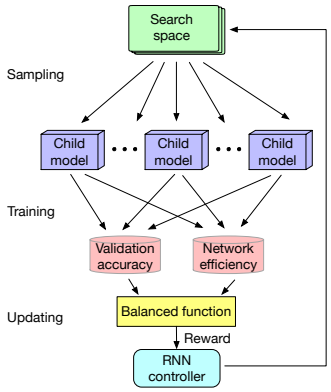


Fig. 2. The balanced-NAS framework. The efficiency and validation accuracy of each child model are coordinated to update the controller.

searching for ideal models. In the self-built radar signal data set, the validation accuracy of the most optimized child model (searching results of NAS in every iteration) could reach up to 92% when SNR is -10 dB, while the number of its trainable parameters is just slightly over half of those based on the standard NAS, e.g., the methods in [7], [9].

In this paper, we proceed with the NAS framework optimization for SEI in different electromagnetic environments. The main contributions are as follows.

1) A feasibility problem is modeled to automatically design the balanced function so that the new design is capable to control the searching result under specific constraints. This overcomes the inflexibility of a fixed predefined balanced function in the early work.
2) The Gaussian process regression (GPR) model is ushered to handle the potential infinite feasible region formed by the constraints which are hard to traverse directly. See Section II for the details of the balanced function and the optimization procedure.
3) The developed technique is evaluated on a self-built and commonly used radar signal data set in modern radar systems [11]. Its excellent capacity is demonstrated by numerical experiments in Section III in designing comparably accurate network structure with far less computation cost compared to the standard NAS methods.

II. Balanced Function Optimization Based on Gaussian Process

A. Block-cell structure and search strategy

The architecture search can follow the "block-cell" mechanism [12]. Let $\mathcal{B}_b$ be the b-th block in a single cell, where $b = 1, 2, ..., n$. A block $\mathcal{B}_b$ (e.g. see $\mathcal{B}_1$ in Fig. 1) composed of five tuples $[I_1, I_2, O_1, O_2, C]$ is a mapping from two input tensors (I_1, I_2) to one output tensor, where $O_1, O_2 \in \mathcal{O}$ are operations applied to the inputs, which contain different types of convolution and pooling, and $C \in \mathcal{C}$ is the action of combining O_1 and O_2, e.g., addition and concatenation. As illustrated in Fig. 1, a cell contains a certain number of blocks, and a full CNN is stacked by several cells.

An LSTM is adopted as a controller to forecast and evaluate the final performance of individual child models. The one-unit fully connected layer following the LSTM embedded layer outputs a series of scalar values representing the probable validation accuracy of the input child models. To overcome the shortcoming within this accuracy-concerned searching strategy and generate the most suitable architectures for various data sets, we proposed a multivariate function, called *balanced function* in [10] (the yellow square in Fig. 2). Compared to the previous search strategy in [7], the proposed framework adds a new metric – network efficiency – to evaluate the performance of child models.

B. Basic properties of the balanced function

The balanced function is denoted by $F(x, y)$, where $x > 0$ and $y > 0$ respectively represent the accuracy and the efficiency of a single network. Its output value is regarded as "scores": the higher the scores, the better the model performs. For analysis convenience, the balanced function is split into two parts, $F_1(x)$ and $F_2(y)$, which denote the accuracy function and efficiency function, respectively. For variable x, we directly use the validation accuracy at every epoch. The variable y – the efficiency of a network – can be measured by the number of trainable parameters of a network in this work.

The balanced function $F(x, y)$ should follow the following two basic properties which reflect the relationship between the accuracy and the efficiency, i.e.,

$$F(x, y_i) > F(x, y_j), \quad \forall y_i < y_j; \tag{1}$$

$$F(x_i, y) < F(x_j, y), \quad \forall x_i < x_j. \tag{2}$$

The first inequality implies that the scores of more complicated models are lower than simpler models corresponding to the same accuracy. The second inequality implies that more accurate models have higher priorities given the same efficiency. Accordingly, $F_1(x)$ and $F_2(y)$ are increasing and decreasing functions, respectively and can be defined by some basic functions, e.g., linear functions, exponential functions,

power functions, and Sigmoid function. The natural way of generating a balanced function using $F_1(x)$ and $F_2(y)$ is

$$F(x,y) = F_1(x)F_2(y) \text{ or } F(x,y) = F_1(x) + F_2(y). \quad (3)$$

C. Feasibility problem for the balanced function

Except for the above basic properties, we suggest considering the following more specific constraints for balanced function generation. I) If a candidate child model i has a higher accuracy and less computation cost compared to model j, the score of model i should be higher than model j obviously. It is however generally hard for a model to get all profits. II) If a minor accuracy promotion leads to a considerable computation cost increase in model i against model j, then model i should be ruled out. In other words, if the accuracy of model j suffers small decrease but its computation cost could reduce more sharply against model i, then model j should be awarded higher scores than model i. III) If there exists a fairly big accuracy improvement for model i, then it should be granted higher tolerance in computation cost.

Overall, we represent these specific constrains regarding balanced function generation as a feasibility problem below:

$$\begin{aligned} \text{find} \quad & F(x,y) && (4)\\ \text{s.t.} \quad & F(x_i,y_i) > F(x_j,y_j), x_i - x_j > 2\lambda; && (5)\\ & F(x_i,y_i) < F(x_j,y_j), x_i - x_j < 2\lambda, y_i - y_j > 2\gamma;\\ & F(x_i,y_i) > F(x_j,y_j), x_i - x_j > 0, y_i - y_j < 0; \end{aligned}$$

where $\lambda, \gamma > 0$ are thresholds for the accuracy and computation cost, respectively.

D. Optimization algorithm based on Gaussian process

It is difficult to directly address the above feasibility problem since the objective function is abstract. Instead, we could initialize $F(x,y)$ based on the aforementioned properties in Eqn (1) and Eqn (2) firstly and then fine-tune it according to these constraints. This way could avoid the waste of searching time in the infinite function space.

The GPR model is a widely-used supervised framework which can make predictions incorporating prior knowledge. Some efficient libraries about the implementation of GPR, nonlinear optimization and hyperparameter tuning have been built in [13]. It assumes that the joint distribution of the training and test data is a multidimensional Gaussian, and thus the predicted distribution can be estimated by the condition on the training data. In this paper, we utilize GPR to fit the point set, which satisfies the above constraints. In detail, firstly, we initialize $F(x,y)$ according to Eqn 1–3. After that, we randomly generate a uniform distribution set $S = \{(x_s, y_s) | s = 1, 2, ...\} \subset \mathbb{R}^{[0,1]}$. Then, substitute all pairs of (x_s, y_s) into $F(x,y)$, and record the pairs of (x_s, y_s) and their corresponding $F(x_s, y_s)$ satisfying the above constraints as $\boldsymbol{X}'$ and $\boldsymbol{Y}'$, respectively. Besides, we divide $\boldsymbol{X}'$ and $\boldsymbol{Y}'$ into three subsets individually, i.e., $\{\boldsymbol{X}'_1, \boldsymbol{X}'_2, \boldsymbol{X}'_3\}$ and $\{\boldsymbol{Y}'_1, \boldsymbol{Y}'_2, \boldsymbol{Y}'_3\}$, where $\boldsymbol{X}'_t$ and $\boldsymbol{Y}'_t$, $t = 1, 2, 3$, record the points satisfying the t-th constraint in Eqn 5. The subset with the minimum scale will be viewed as the training data for the Gaussian process, denoted as $\boldsymbol{X}$ and $\boldsymbol{Y}$. Therefore, the posterior distribution over the unknown output $\boldsymbol{Y}_*$ given the test input $\boldsymbol{X}_*$ reads

$$p(\boldsymbol{Y}_* | \boldsymbol{Y}) = \mathcal{N}(\boldsymbol{\mu}_{GP_*}, \boldsymbol{\sigma}^2_{GP_*}), \quad (6)$$

which is Gaussian with the following mean and variance:

$$\boldsymbol{\mu}_{GP_*} = \kappa_{\mathrm{f}*}^\top (\kappa_{\mathrm{ff}} + \sigma^2 \boldsymbol{I})^{-1} \boldsymbol{Y}, \quad (7)$$

$$\boldsymbol{\sigma}^2_{GP_*} = \boldsymbol{\sigma}^2 + \kappa_{**} - \kappa_{\mathrm{f}*}^\top (\kappa_{\mathrm{ff}} + \sigma^2 \boldsymbol{I})^{-1} \kappa_{\mathrm{f}*}, \quad (8)$$

where $\kappa(\cdot)$ is the kernel function, $\kappa_{\mathrm{f}*}$ contains the kernel similarities of the test points $\boldsymbol{X}_*$ to all training points, κ_{**} is the self-similarity of $\boldsymbol{X}_*$, κ_{ff} contains the kernel similarities of the training points, $\boldsymbol{\sigma}^2$ is the hyperparameter accounting for the variance in the training data, and $\boldsymbol{I}$ is the identity matrix. The trick of adding some random disturbance into the training data set could increase samples diversity.

III. Experiments

A. Experimental setup

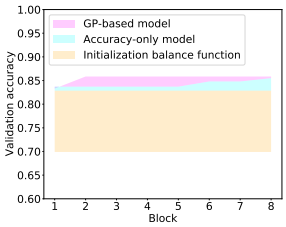
The radar signal data used is a self-built data set "SIGNAL-5" used in [10] referring to previous research works e.g. [14]. It contains 7 different radar signals (LFM with positive and negative slope, sine FM, 64 bit polyphase code signals including P1, P2, P3 and P4) and 5 different electromagnetic environments (SNR set to -10dB, -8dB, -5dB, -1dB and noise free). We utilize STFT to transform the raw time-series radar signal to the time-frequency distribution images and clip them to the size of 64×64 using the nearest neighbor interpolation before feeding them into CNN for the sake of computation reduction. The hidden state size and the number of cells in the LSTM controller are both 100. The Adam optimizer is utilized with learning rate 10^{-3} initially and then exponential decay. There are two types of CNN cells (purple part in Fig. 1) which have the number of filters 32 and 64 with stride set to 2. The learning rate for the full CNN is 10^{-3} initially with cosine decay. To construct the full CNN from cells, there is a global average pooling layer before the final dense layer. The sigmoid function and exponential function are used to initialize the balanced function $F(x,y)$, i.e.,

$$F_1(x) = \frac{1}{1 + e^{-10(x-0.5)}}, \quad F_2(y) = e^{-y}, \quad (9)$$

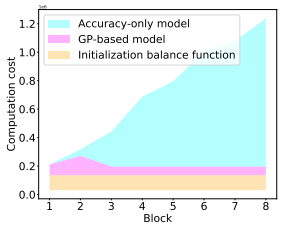
$$F(x,y) = \frac{1}{2}[F_1(x) + F_2(y)]. \quad (10)$$

B. Results and analysis

We firstly compare the performance between the proposed GP-based NAS (*GP*), NAS with initial balanced function (*ini*) and accuracy-only NAS (*acc-only*). Specifically, we utilize the progressive NAS proposed in [9] as the acc-only NAS baseline. Here, we choose the data set with SNR = -10dB and set the thresholds for the accuracy and computation cost to $\lambda = 0.03$ and $\gamma = 0.02$. The result is shown in Fig. 3. The region indicates the top-10 candidate models among all searching results. Obviously, the proposed GP-based NAS method achieves the highest accuracy with acceptable

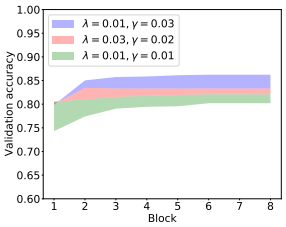


(a) Validation accuracy

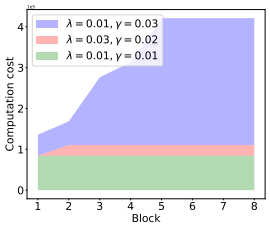


(b) Computation cost

Fig. 3. Performance comparison in terms of accuracy (a) and computation cost (b) between the proposed GP-based, ini and acc-only NAS methods. Each colorful area in the figures show the upper and lower bounds of the performance of the top-10 candidate models selected by each method against different numbers of blocks.



(a) Validation accuracy



(b) Computation cost

Fig. 4. Performance of the proposed GP-based NAS method in terms of accuracy (a) and computation cost (b) corresponding to different λ and γ in Eqn 5. Smaller λ values would stimulate the concentration of the GP-based method more on the accuracy, while smaller γ values could encourage the GP-based method to find more efficient neural structures.

computation cost. In particular, the average validation accuracy of the three methods achieved is $x_{\mathrm{GP}} = 85.5\%$, $x_{\mathrm{ini}} = 82.9\%$ and $x_{\mathrm{acc-only}} = 84.2\%$, respectively. The average number of trainable parameters of the models the three methods selected is $y_{\mathrm{GP}} = 2.1\times10^5$, $y_{\mathrm{ini}} = 1.4\times10^5$ and $y_{\mathrm{acc-only}} = 7.2\times10^5$, respectively. Compared to the proposed algorithm, the acc-only NAS could reach a competitive accuracy but also lead to an explosion of computation cost. Furthermore, if only utilizing the initialized balanced function in Eqn 10 without optimization, most of effort will be made to decrease the computation cost, while the accuracy promotion is prone to be ignored (the accuracy keeps unchanged as blocks increasing).

Since the constraints on the validation accuracy and computation cost could be adjusted regarding specific requirements, another advantage of the proposed GP-based method is that it significantly enhances the flexibility of the conventional NAS. The performance of the proposed GP-based method corresponding to different pairs of $[\lambda, \gamma]$ is shown in Fig. 4. For example, to put more stress on the accuracy, λ should be set to a smaller value so that a small accuracy increase would bring obvious augmentation of the scores. As a result, there will be a preference for more accurate models as the searching procedure ongoing. Moreover, if the efficiency of the network is more important for some specific tasks, we could choose a smaller γ value so that scores of child models would drop albeit their computation cost only goes up a bit.

IV. Conclusion

In this paper, a novel NAS mechanism, GP-based NAS, was proposed for SEI, bringing the benefit of automatically finding the most optimized model satisfying specific constraints. The mechanism is based on the Gaussian process regression to conducting balanced function optimization considering both the validation accuracy and computation cost. It provides an access to adjust the choice of model styles according to specific requirements, e.g. either imposing more stress on the accuracy or efficiency in NAS. Experimental results demonstrated that the proposed GP-based mechanism could design models with the highest accuracy and over three times less trainable parameters compared to the standard NAS. Moreover, the proposed mechanism can leverage the searching mechanism to find the fittest model by changing the accuracy or efficiency threshold, significantly improving the versatility and flexibility of the NAS.

References

[1] Kenneth I Talbot, Paul R Duley, and Martin H Hyatt, "Specific emitter identification and verification," *Technol. Rev.*, vol. 11, pp. 113–133, 2003.

[2] Guillaume Revillon, Ali Mohammad-Djafari, and Cyrille Enderli, "Radar emitters classification and clustering with a scale mixture of normal distributions," *IET Radar, Sonar Navig.*, vol. 13, no. 1, pp. 128–138, 2018.

[3] Qiang Guo, Xin Yu, and Guoqing Ruan, "Lpi radar waveform recognition based on deep convolutional neural network transfer learning," *Symmetry*, vol. 11, no. 4, pp. 540–553, 2019.

[4] Thomas Elsken, Jan Hendrik Metzen, and Frank Hutter, "Neural architecture search: A survey," *J. Mach. Learn. Res.*, vol. 20, no. 55, pp. 1–21, 2019.

[5] Mingyang Du, Ping Zhong, Xiaohao Cai, and Daping Bi, "Dncnet: Deep radar signal denoising and recognition," *IEEE Trans. Aerosp. Electron. Syst.*, pp. 1–1, 2022.

[6] Xin He, Kaiyong Zhao, and Xiaowen Chu, "Automl: A survey of the state-of-the-art," *Knowl. Based Syst.*, vol. 212, pp. 106622, 2021.

[7] Barret Zoph and Quoc V. Le, "Neural architecture search with reinforcement learning," in *Int. Conf. Learn. Represent., ICLR*, 2017, pp. 1–16.

[8] Esteban Real, Alok Aggarwal, Yanping Huang, and Quoc V. Le, "Regularized evolution for image classifier architecture search," in *33rd. AAAI Conf. Artif. Intell.*, 2019, pp. 4780–4789.

[9] Chenxi Liu, Barret Zoph, Maxim Neumann, Jonathon Shlens, Wei Hua, Li-Jia Li, Fei-Fei Li, Alan Yuille, Jonathan Huang, and Kevin Murphy, "Progressive neural architecture search," in *Lect. Notes Comput. Sci.* Springer, 2018, pp. 19–34.

[10] Mingyang Du, Xikai He, Xiaohao Cai, and Daping Bi, "Balanced neural architecture search and its application in specific emitter identification," *IEEE Trans. Signal Process.*, vol. 69, pp. 5051–5065, 2021.

[11] Thokala Ravi Kishore and K Deergha Rao, "Automatic intrapulse modulation classification of advanced lpi radar waveforms," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 53, no. 2, pp. 901–914, 2017.

[12] Zhao Zhong, Junjie Yan, Wei Wu, Jing Shao, and Cheng-Lin Liu, "Practical block-wise neural network architecture generation," in *Proc. IEEE Comput. Soc. Conf. Comput. Vision Pattern Recognit.*, 2018, pp. 2423–2432.

[13] Ruben Martinez-Cantin, "Bayesopt: a bayesian optimization library for nonlinear optimization, experimental design and bandits.," *J. Mach. Learn. Res.*, vol. 15, no. 1, pp. 3735–3739, 2014.

[14] Ming Zhang, Lutao Liu, and Ming Diao, "Lpi radar waveform recognition based on time-frequency distribution," *Sensors*, vol. 16, no. 10, pp. 1682–1701, 2016.