S11 Parameter Calculation of Frequency Selective Surface Based on Deep Learning

To cite this article: Xi Ren *et al* 2021 *J. Phys.: Conf. Ser.* **1865** 042022

View the article online for updates and enhancements.

S11 Parameter Calculation of Frequency Selective Surface Based on Deep Learning

Xi Ren[*], Changlin Liu and Minghui Zeng

National Key Laboratory of Science and Technology on Micro/Nano Fabrication, Department of Micro/Nano Electronics, School of Electronic Information and Electrical Engineering, Shanghai Jiao Tong University, Shanghai, 200240, PR China

*Corresponding author: SJTU_RX@sjtu.edu.cn

Abstract. This paper proposes the FSS-CNN network model as a forward predictor, replacing the function of the Maxwell equation solver of commercial software. The predictor is different from the numerical optimization method, and the data-driven method based on machine learning (ML) can be expressed and generalize complex functions or data to discover unknown relationships between a large number of variables. In the frequency range of 2~18GHz, the S11 parameter prediction of the corresponding metal pixel pattern can be easily realized by an accurate forward neural network model. The MSE reaches the level of less than 0.1 and the time consumption is less than 0.07s, which meets the requirements of fast, efficient and automatic calculation.

Keywords: Pixel FSS, Deep Learning, S11 Parameters, EM Solver.

1. Introduction

Deep learning has been widely used in various fields in recent years, as is the field of electromagnetic computing. Iterative updates of deep learning algorithms continue to push the limits of traditional image, voice and video recognition and processing, thereby greatly promoting the development of modern machine learning technology. At the same time, it began to penetrate into other disciplines, such as biology, genetics, materials science and physics [1]. Deep learning includes discriminant model and generative model. Neural network is a kind of discriminant model. It has been applied to solve some prediction problems in electromagnetics [2-5]. However, due to the shallow structure, the representation ability is poor and the effect is limited. Some recent work has proposed deep neural networks to model nanophotonic metamaterials [6, 7]. However, the network model based on multiple fully connected layers can only handle the simple structure of optical and electromagnetic response prediction problems. The proposal of convolutional neural network makes up for this shortcoming to a certain extent. As the most successful direction of deep learning application, convolutional neural network extracts abstract and non-abstract features of input data layer by layer through operations such as convolution, pooling, and dropout [8, 9]. It uses data-driven methods to represent and generalize complex functions or data models to discover unknown relationships between a large number of variables. This article is mainly based on the CNN model of deep learning theory. Aiming at the S parameter calculation problem of single-channel pixelated FSS, this paper proposes deep convolutional network model prediction to realize the prediction function of commercial electromagnetic computing software. The network model process is shown in Figure 1-1.

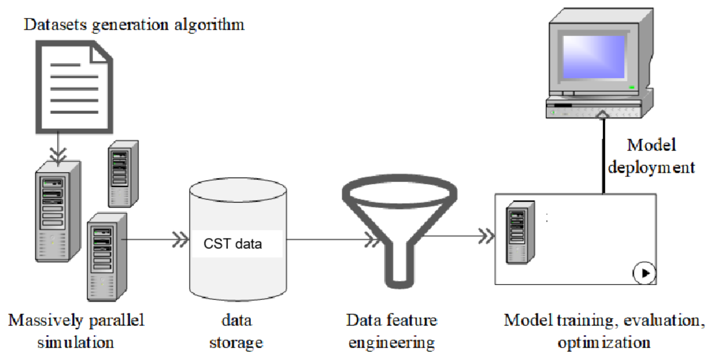
Datasets generation algorithm
CST data
Model deployment
Massively parallel simulation
data storage
Data feature engineering
Model training, evaluation, optimization

Fig. 1-1 Deep learning model training process

First, the model is positioned as a regression problem. Figure 1-1 shows that multiple types of data are obtained through CST numerical calculation, and defines the training data set used for FSS pixel map prediction and generation model. First, the traditional deep learning-based forward convolutional neural network can establish the complex relationship between the FSS metal pixel point distribution and its electromagnetic response from many pixel patterns and the corresponding S parameters in the 2~18GHz frequency band, and change the structure value Representation. In the case of a certain metal pattern, the most reliable calculation method to obtain accurate electromagnetic response results is numerical simulation through commercial electromagnetic calculation software, which consumes huge computing resources. Deep learning data-driven models can be used to study complex input and output interactions to inspire us to use deep learning or machine learning design methods for FSS calculation research. Therefore, compared with numerical calculation methods, based on the application research of deep learning (DL) in the fields of image, speech, and materials, due to the huge advantages of pixelated FSS surface coding, the coding pattern of the unit can be established from physical space to numerical value using deep learning methods. Spatial feature mapping greatly simplifies the workload of the network, thereby avoiding time-consuming numerical calculations to solve Maxwell equations. The model uses CST simulation data to train and test the deep learning model. Finally, we deploy the trained model online to enable it to implement effective experimental research applications on the local workstation.

2. Experimental method

2.1. FSS-CNN network framework design

In the field of artificial intelligence, deep learning has great advantages in the feature extraction of images. The predictor in this section is designed based on the theory of Deep Convolutional Neural Network (CNN). The goal is to accurately solve pixelation. S parameter of FSS. For the image shape input, the features are first extracted, and a network model describing the relationship between the patch pattern and the S parameter is reconstructed from the image through several features, as shown in Figure 1-2a. The hidden information of binary patterns is different from simple geometric patterns. It is difficult for the human eye to discover the distribution of a large number of pixels and the relationship between the input pattern and the target space. However, the deep learning network can extract the features of the pattern from the bitmap through the convolutional layer, and change the receptive field by changing factors such as the size and step length of the convolution kernel, zooming in and closer to the image, and extracting the effective features. The filtering discrimination is very low. Characteristics. The features generated after convolution of the spatial information of each layer are input to the next layer of network, and the weight of each layer of network is trained through reverse derivation. In order for this network model to accurately describe this relationship, it is necessary to learn a large number of samples to improve the expressive ability of the CNN model, so that the accuracy of the CNN model to unknown data is improved.

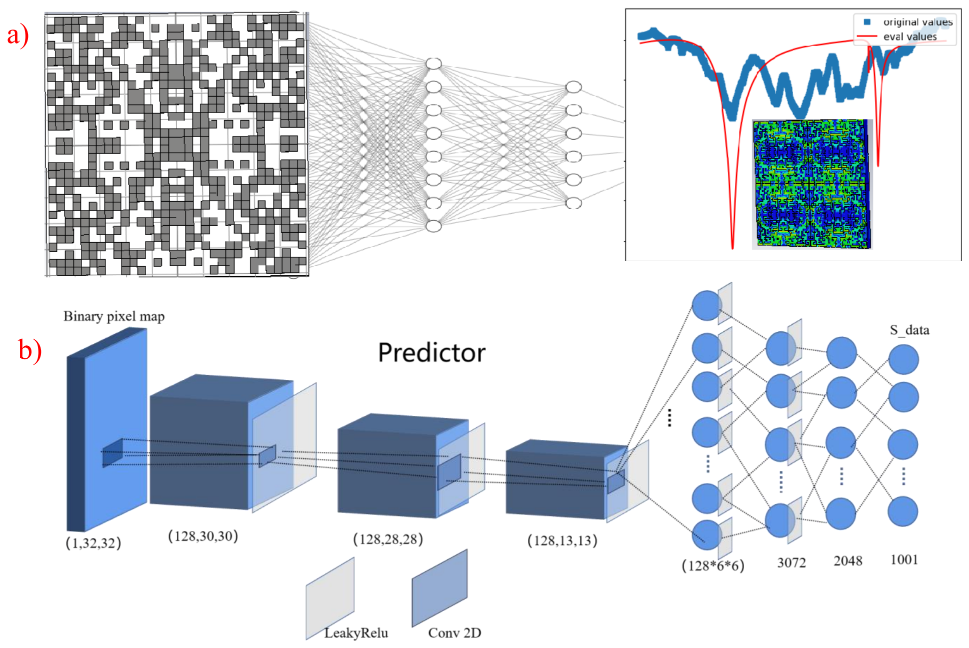

Fig. 2-1 Network structure of the FSS-CNN model

The network structure is shown in Figure 1-2b. After the 32*32*1 binary image is extracted from the original data features by the conv2D layer, the number of data channels is increased to 128, and the information storage space is rapidly increased, and then continues through three layers of convolution Operation and two-layer full connection make the data features compressed to 1004 target points. The full-time matrix of each layer uses Xvmair to initialize its distribution, making it easier to converge spatially. Because the input data is small, 0 or 1, the problem of returning the gradient to zero when backpropagating the derivation is easy, and the target result has a negative number with a large absolute value, so use the LeaklyRelu activation function after each layer. The ReLU function can better solve the "ReLU" problem.

2.2. Datasets

In this study, the properties of the FR-4 board were kept fixed, and the randomly coded metal pattern was used as the only variable of the passive FSS. Deep learning is a typical supervised learning. The input data of the network is a set of 32*32*1 binary images, and the label data is 1*1001 S parameters. The code and frequency spectrum form a set of tag data that is mapped to each other. In other words, these codes constitute the sample set of the entire parameter space, and the S spectrum is the sample set of the response space. The purpose is to express the response from the complete design space to the complete response space by establishing a mutual mapping relationship between two sample sets. In order to implement this algorithm, first of all, a new scheme for better preparing the training data set needs to be proposed. However, the pixelated encoding method has a disadvantage that cannot be ignored. It generates too much invalid data. The binary pixel map has a computational complexity. Many discretized pixels cannot produce an effective surface current response to incident electromagnetic waves and do not constitute a frequency. Select the condition of the surface element. Therefore, the pattern search must be restricted. In the pixel code representation, the sub-blocks with and without metal represent "1" and "0" respectively at the microscopic level. In Figure 2-2, yellow pigment points and dark pigments are used. Point representation.

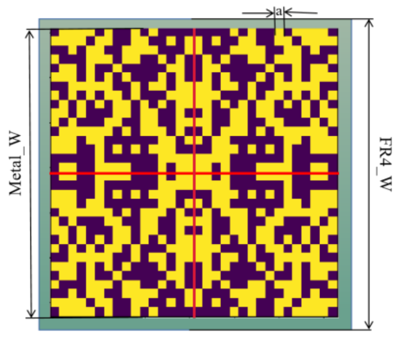

Fig. 2-2 Pixelated metal image unit

It can be seen from the above that the unit pattern is divided into 32 × 32 grids. When only the "01" code is considered, the maximum total number of pattern possibilities for each method is shown in Table 3-1. According to the scheme, a total of 6000 different 01 matrices are initially generated. Then, use these pixel distributions to generate a metal layer, set the thickness to 0.03mm, set the metal pattern side length Metal_W to 10mm, and FR4 board side length FR4_W to 10.2mm. The pixel points are divided into, pixel size is 0.3125mm, FR4 thickness Set to 1mm, use commercial EM software CST2020 to simulate the S parameter data of 2~18GHz linearly polarized plane waves incident perpendicularly along one side of the metal pattern. Finally, by adding random noise to the pixel image, a total of 6000 samples for the predictor training set are formed, in which the binary pixel image is used as the input data, and the corresponding S parameter data is the output data. The training data set is stored in the laboratory On the internal server.Three servers are simulated in parallel, which takes 170 hours. Finally, the data set is split into three: training set, validation set, and test set. Train the model on the training set. Test the model on the validation set to ensure that there is no overfitting. Select the model with the best comprehensive performance, test it on the test set, and report the true accuracy of the model.

2.3. Model training and optimization

The training deep learning model in this experiment uses computing resources for three win10-OS PCs equipped with NVIDIA GeForce RTX2080Ti, Intel(R)Core (TM) i7-9800X CPU@3.8GHz and 32GB RAM. The training model is based on Pytorch and Sklearn frameworks. Python programming language is built. Model training and evaluation is a mathematical optimization problem. The model can predict continuous S parameter values instead of discrete labels, so the last two fully connected layers of the network are linear layers without activation. The model P can theoretically learn to predict values in any range. The mean square error (MSE) between the network output and the actual S parameter is introduced as the loss function of the prediction model, which is defined as the square of the difference between the predicted value ($\hat{S}$) and the EM simulation value (S) to evaluate the convergence of the model. Because the loss function is composed of input parameters and inherent network weights, in the model training stage, the input parameters are used as weights and the network parameters are identified as variables, so as to transform the training process into convex optimization of the loss function.We randomly select 80% of the data as the training set and 20% of the data as the validation set to test the prediction accuracy of the model. MSE is used as the loss function of the predictor. Use Adam optimizer as an optimizer for initial training.

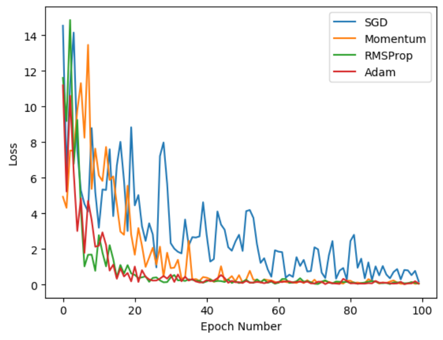

Fig. 2-3 Comparison of the effects of different optimizers

The hyperparameter setting of network training has an important influence on the convergence speed of the network, especially the choice of learning rate. The learning rate is a hyperparameter that guides us on how to adjust the weight of the network through the gradient of the loss function.The experimental comparison results of the optimizer are shown in Figure 2-3.In this section, we choose 0.001 for the initial value of the learning rate to make the initial convergence faster. For other parameters other than the learning rate of Adam, see Table2-1 for the initial setting of the batch amount.

$$new\_weight = last\_weight - lr\_rate \times gradient \tag{2-1}$$

Table. 2-1 Hyperparameter initial value setting

Learning_rate	1e-3
Bathch size	10
weight_decay	1e-2
Regularization	L2

In this section, we design experiments, select table2-1 hyperparameters, and set the maximum training round to 300. Assuming that the learning rate decreases over time, the model can finally converge. However, it is found that when lr=0.001, after 20 rounds of training, the loss stabilizes at around 11.0 and does not continue to decrease. It may be more difficult to improve the training loss as the gradient reaches a plateau.

Save the model and interrupt the training. Try to increase the momentum parameter of the Adam optimizer. lr gives up the fixed value and uses the dynamic value. Pytorch provides six learning rate adjustment methods, which can be divided into three categories: orderly adjustment, adaptive adjustment, and custom adjustment. The first category is to make adjustments in an orderly manner according to certain rules. This category is the most commonly used. They are equal interval descent (Step), set descending interval (MultiStep) on demand, exponential descent (Exponential), and CosineAnnealing. The timing of adjustment of these four methods is artificially controllable and is also commonly used during training. The second type is to adjust according to the training situation. This is the ReduceLROnPlateau method. This method monitors the change of a certain index. When the index does not change much, it is the time to adjust the learning rate, so it is an adaptive adjustment. The third category, custom adjustment, Lambda. The adjustment strategy provided by the Lambda method is very flexible. We can set different learning rate adjustment methods for different layers. This is very useful in fine-tune. We cannot only set different learning rates for different layers, but also It sets different learning rate adjustment strategies. Therefore, by introducing the adaptive rate adjustment method optim.lr_scheduler, when the loss of five consecutive rounds of loss is less than 1%, the learning rate is increased to the original 1.1, and then the model adaptively adjusts the learning rate according to the ReduceLROnPlateau method, and prints the learning rate in real time Change the state, actively terminate the training when necessary, and manually adjust the lr.

Finally, in the predictor training process, after 260 rounds of exploratory training on 3000 data sets, there was a huge gap between the training set loss value and the test loss value. The fitting effect of the training set was good, train_loss dropped to less than 0.03, and eval_loss was Oscillation near 2.0, the results are shown in Figure 2-4, indicating that the generalization effect of the model is relatively poor, and the data fitted on this basis will lead to poor accuracy of the test set of the model. There may be two situations. One is that the number of data sets is too small to reflect the overall sample distribution space. The second is that the over-fitting phenomenon occurs during the model training process. Through L2 and L1 regularization and adjusting the Droupout layer respectively, this phenomenon is reduced. It shows that the pixelated metal pattern does have a spatial mapping relationship with the S-parameters, but there is limited room for improving the model's capabilities, and the data set generation scheme should be improved. Therefore, by increasing the total number of data sets to 5000, 7000, 10000, and increasing the proportion of data generated by the 1/4 random method scheme to 50%, the two improvements make it possible to appropriately reduce the degree of freedom of the data set while increasing the sample space. According to Figure 2-5, the average error value of the predictor is negatively correlated, indicating that under the condition of 32*32-pixel division accuracy, the data set of 3000 is much smaller than the total number of samples, and the model will not have enough training samples, resulting in Its test accuracy is low. Therefore, it is necessary to improve the data set division and generation plan, reduce the solution space, and ensure that the training set can cover enough sample distribution space under the existing computing power, which can improve the generalization ability of the model and the actual deployment accuracy.

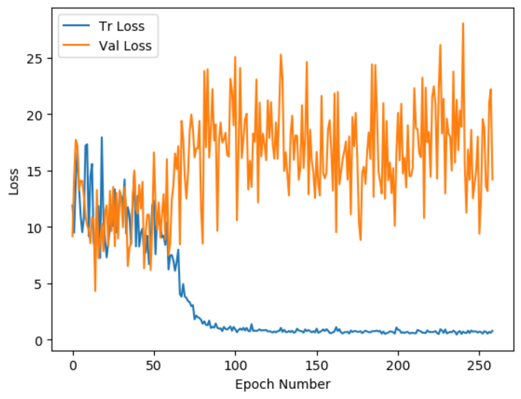

Fig. 2-4 Small-scale data set training set loss and validation set loss

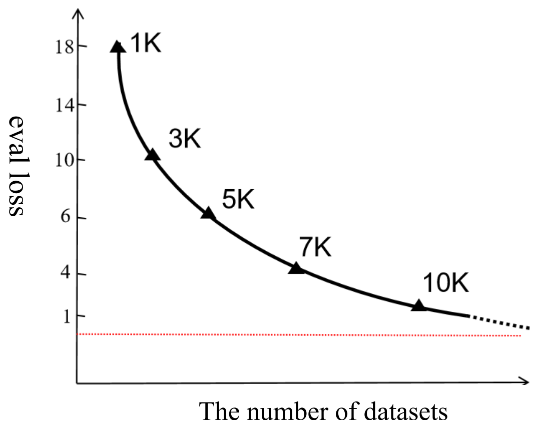

Fig. 2-5 Impact of data set sample size on model accuracy

When eval_loss and train_loss also converge to less than 0.5, we test the divided 1000 test set samples and compare the S parameters of EM numerical simulation with the S value predicted by the predictor P. The predictor can calculate the input in only 0.8s The S parameter of the FSS metal pixel map is very efficient. Figure 3-13 is the 9 sets of prediction regression accuracy maps. The black circle represents the simulated reflection phase, and the red asterisk represents the predicted reflection phase. We can see that the black circles and red stars are almost coincident in each group. It can be seen that the accuracy of the predictor P is not only better for single resonance peak feedback, but also the prediction effect of the multi-pass filter is better. It is a poor fit for narrowband absorption peaks <300MHz.

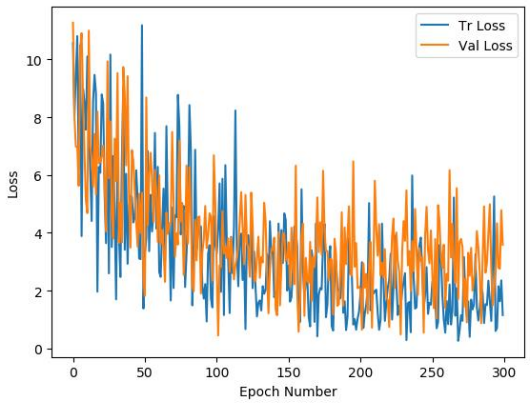

Fig. 2-6 Increase the data set to 3000 training set and validation set loss

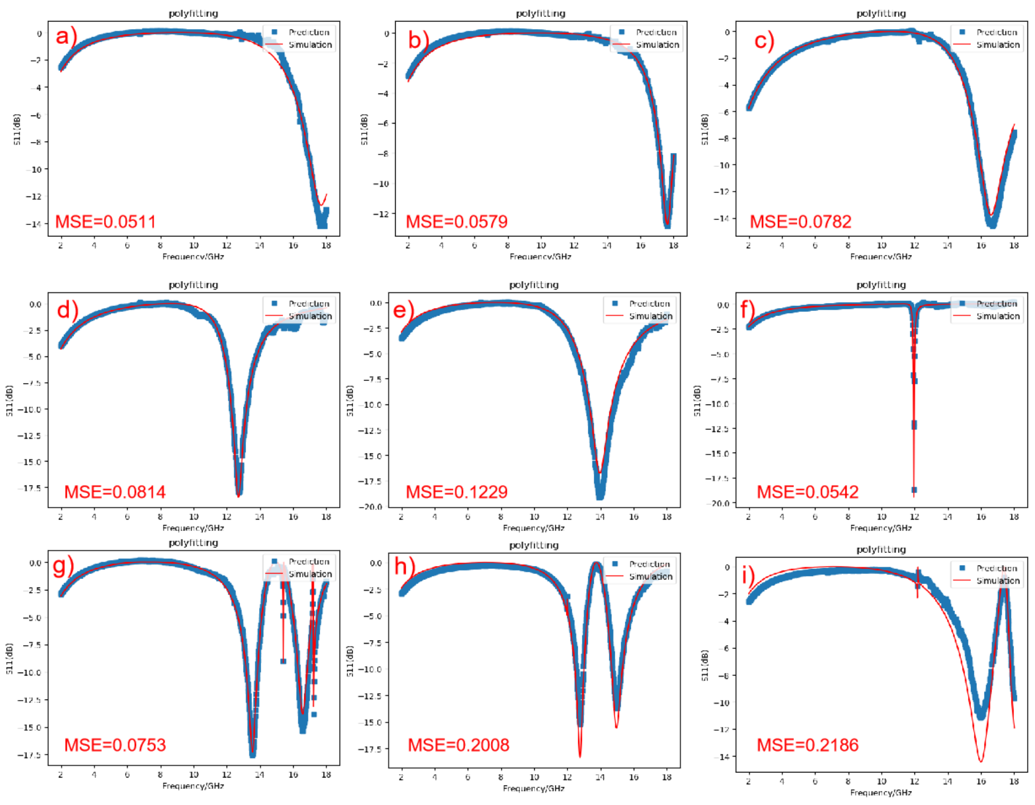

Fig. 2-7 The predictor calculates the results of 9 different types of validation set and test set samples S11

3. Conclusions

The predictor undergoes a comparative experiment of four types of optimizers through the model. The Adam optimizer is selected. The learning rate is set to 1e-3. After 300 rounds of training, Train_loss and eval_loss drop to less than 0.5, and the correlation coefficient is greater than 0.87. The predictor should It can be calculated more accurately. Manually adjust the learning rate, and finally after 100 rounds of training, it is found that the validation set loss can continue to drop to less than 0.1. Load the trained model directly according to the results of the verification set. Figure 2-7 shows the test results of 9 sets of geometric samples from the test set. The average calculation time is 0.07s. The prediction curves and simulation curves in each group almost overlap. The simulation results show good agreement. It shows that the model has good practical value in the ultra-wideband frequency of 2-18 GHz.

Acknowledgments
This work has no financial support from anyone.

References

[1] Wei Hui, Pan Yunhe. From Knowledge Representation to Representation: Advances in Artificial Intelligence Epistemology [J]. Computer Research and Development, 2000(07): 819-825.
[2] Kabir H, Wang Y, Yu M, et al. Neural Network Inverse Modeling and Applications to Microwave Filter Design [J]. IEEE Transactions on Microwave Theory & Techniques, 2008,56(4):867-879.
[3] Freitas G, Rego S L, Vasconcelos C. Design of metamaterials using artificial neural networks: 2011 SBMO/IEEE MTT-S International Microwave and Optoelectronics Conference (IMOC 2011), 2011 [C]. IEEE.
[4] Vasconcelos C F, Rêgo S L, Cruz R M. The use of artificial neural network in the design of metamaterials: International Conference on Intelligent Data Engineering and Automated Learning, 2012 [C]. Springer.
[5] Capizzi G, Lo Sciuto G, Napoli C, et al. A multithread nested neural network architecture to model surface plasmon polaritons propagation [J]. Micromachines, 2016,7(7):110.
[6] Malkiel I, Nagler A, Mrejen M, et al. Deep learning for design and retrieval of nano-photonic structures [J]. arXiv preprint arXiv:1702.07949, 2017.
[7] Peurifoy J, Shen Y, Jing L, et al. Nanophotonic particle simulation and inverse design using artificial neural networks [J]. Science advances, 2018,4(6):r4206.
[8] Peurifoy J, Shen Y, Jing L, et al. Nanophotonic particle simulation and inverse design using artificial neural networks [J]. Science advances, 2018,4(6):r4206.
[9] Romero M, Interian Y, Solberg T, et al. Training Deep Learning models with small datasets [J]. arXiv preprint arXiv:1912.06761, 2019.