Semi-Supervised Accuracy Predictor: SemiLGB

Hai Li*
National Key Laboratory of Multi-spectral Information Processing Technology, Huazhong University of Science and Technology
Wuhan, China
darksosol26@163.com

Yang Li*
Research Institute of Robotics, Shanghai Jiaotong University
Shanghai, China
shjt_ai@163.com

Zhengrong Zuo*†
National Key Laboratory of Multi-spectral Information Processing Technology, Huazhong University of Science and Technology
Wuhan, China
zhrzuo@hust.edu.cn

Abstract

The effect of neural architecture search (NAS) depends on accuracy predictor. However, the NAS needs abundant and high-quality pairs of architectures and their accuracies, while it is expensive to evaluate an architecture for obtain its accuracy. Aiming at the accuracy predictor for the few-shot, SemiLGB method is proposed, which uses different types and unlabeled architectures. Specifically, SemiLGB 1) uses the re-weighting method to train the initial accuracy predictor in small set of different types of architecture-accuracy data pairs with the feature engineering of data mining through LightGBM; 2) uses the trained accuracy predictor to predict the accuracy of the unlabeled architectures; 3) adds the generated data pairs which are selected by semi-supervised search to the original data to further improve the predictor. 4) uses the model deviation correction to improve the predictor performance which is based on the re-weighting method. The trained accuracy predictor can be applied to various NAS algorithms by predicting the accuracy of candidate architectures for them. SemiLGB has two advantages: 1) Compared with the network, the tree model which is white box training method is more stable in few-shot scenario. 2) Training LightGBM in low resource environment on CPU rather than GPU, can ensure accuracy and efficiency. On the CVPR2021-NAS performance estimation track, SemiLGB achieves 0.189 RMSE. We further apply the proposed method to NN model, which also improves the NN performance.

1. Introduction

NAS usually consists of two components: one controller controls the generation of new architectures, and the other evaluator trains candidate architectures and evaluates their accuracies. In order to ensure the performance of the evaluator, a large number of high-quality architecture-accuracy pairs are needed. However, it is costly to collect such data, because it is expensive for evaluator to train each architecture for obtaining its accuracy, which results in the highest computational cost in NAS. Popular methods usually consume hundreds to thousands of GPU days to find a good architecture [1, 2].

In order to solve this problem, one shot NAS [3, 4, 5] uses a supernet, which includes all candidate architectures through weight sharing, and trains the supernet to reduce the training time. At the same time, the quality of the training data (Architectures and their corresponding accuracies) of the controller will decrease. In the case of limited labeled training data, semi-supervised learning is a popular method to improve training accuracy by using unlabeled data. In NAS, unlabeled architectures can be obtained by randomly generating, mutating [2], or simply traversing the entire search space, which almost does not incur any additional cost. SemiNAS [1] proposes a semi-supervised learning method to improve the performance of NAS search.

Inspired by semi-supervised learning, We propose SemiLGB method, which can train the data with different types and unlabeled architectures. With four improvements to enhance the diversity of features, utilization of data and regression ability of model. Specifically, SemiLGB 1) uses the re-weighting method to train the initial accuracy predictor in small set of different types of architecture-accuracy data pairs with the feature engineering of data mining through LightGBM [6]; 2) uses the trained accuracy predictor to predict the accuracy of the unlabeled architectures; 3) adds the generated data pairs which is selected by semi-supervised search to the original data to further improve the predictor. 4) uses the model deviation correction to improve the predictor performance which is based on the re-weighting method. The trained accuracy predictor can be applied to various NAS algorithms by predicting the accuracies of candidate structures for them.

Compared with the traditional neural network accuracy predictor, SemiLGB based on LightGBM model has better generalization performance in few-shot scenario. All the methods include feature engineering、 semi-supervised training、re-weighting and model deviation correction can be effectively applied to the different types of models. The main contributions of the paper can be summarized as follow.

• Proposing a new semi-supervised method, and the re-weighting method is designed to fuse data sets with

different types of architectures.

• Designing the feature engineering method to strengthen important model architecture features which has good robustness and explicable.

• Presenting the model deviation correction method to improve the original predictor.

• Proving the above parts can achieve effective improvement in tree model and NN model.

2. Related Work

From the perspective of the computational cost of training candidate architecture, the previous research on NAS can be divided into conventional NAS and one-shot NAS. Conventional NAS include [1, 2], which have achieved significant improvements on multiple benchmark datasets. In conventional NAS, it is expensive to obtain the accuracies of candidate architectures, because they train each architecture from scratch and usually require thousands of architectures to train. The total cost usually exceeds hundreds of GPU days [1, 2].

In order to reduce the huge cost of NAS, the one-shot NAS based on weight sharing mechanism is proposed [1, 3]. It is suggested that all candidate operations should be included in the search space of supernet, and parameters should be shared among candidate architectures. Each candidate architecture is a sub-graph in the supernet, and only relevant parameters are activated. The algorithm first trains the supernet, and then evaluates the accuracies of candidate structures according to the corresponding sub-graphs in the supernet [1, 4, 5]. While using different search algorithms, it also uses the idea of one-shot to perform efficient search. This weight sharing mechanism successfully reduces the calculation cost to less than 10 GPU days [1, 4, 5]. However, supernet needs careful design and training. In addition, compared with conventional NAS, its performance and reproducibility are poor. One of the main reasons is that the training time is short and the update of a single architecture is insufficient, which leads to the inaccurate ranking of the architecture and provides the controller with a relatively low quality architecture-accuracy pairs. SemiNAS achieves good results through semi-supervised learning and lightweight search [1]. Its disadvantage is that it directly adds the predicted samples to the training set, without sampling methods such as weight to distinguish the effectiveness of samples. At the same time, the Imagenet data set used for SemiNAS is still large, which can not prove its performance in few-shot scenario.

3. SemiLGB

In this section, we describe the main parts of the SemiLGB's pipeline as follow.

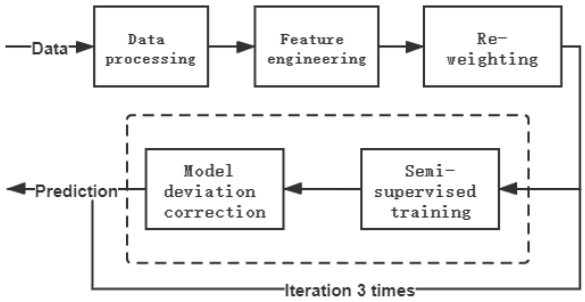

Figure 1: The pipeline of SemiLGB method.

3.1. Data Processing

The CVPR2021-NAS performance estimation track's data set is based on the Mobilenet-like search space, each sample contains 16 features after simplification. Each feature corresponds to the kernel connection mode of one block, which has total 6 kernel modes. The train set consists of 31 samples of nonlinear topology network architecture and 200 samples of linear topology network architecture, the test set has 100,000 samples of nonlinear topology network architecture. The binary class feature is added whether it is a linear topology as 17th feature.

For each label y' of linear topological network architecture, transform the label to y as follow.

$$y = \frac{mean2}{mean1} * y' \tag{1}$$

Where $mean1$ is the mean value of the label of linear topological network architecture, $mean2$ is the mean value of the label of nonlinear topological network architecture. This can reduce the label gap of two architecture types.

Considering the decision tree splitting [6] for the LightGBM model, in k-fold cross validation, the value is replaced by nan that does not appear in the train data but in the valid data to improve the generalization ability.

3.2. Feature Engineering

Based on the feature engineering method of data mining, according to the feature importance of decision tree, several most important features are as follow:

• The most common kernel mode in the sample.

• The 16 features of blocks are divided into four adjacent groups, each of which has the most common kernel mode.

• The features of adjacent blocks are subtracted.

In addition to the above features, we also design:

• Combined count feature and proportion feature.

• Division feature.

• One-hot encoding.

These features can be set up to be generated automatically to deal with new data sets for the kernel mode feature. Even for the NN model which can automatically extract features, using data after feature engineering can get

better performance compared with the original data.

3.3. Re-Weighting

The data of nonlinear topology network architecture is divided into 5-fold cross validation. The data of linear topology network architecture is added to each fold train data, and the sample weight is set to 0.05. At the same time, the semi-supervised data set is added to each fold train data, and the sample weight is set to 0.01. The semi-supervised data set is obtained from the predicted test set by semi-supervised search. After the above settings, the more data can be used to improve the predictor. The weight of the train set with nonlinear topology is set based on the count feature. The specific weight formula of each sample is as follows:

$$weight = \frac{\sum_{i=1}^{16}(1/count(i))^{pow}}{mean} \quad (2)$$

Where $count(i)$ represents the count number of corresponding features of the sample, and $mean$ represents the mean value of all samples' $weight$, namely, the average weight is set to 1, and pow is taken as 5 through experiments. After such a setting, the long tail distribution of the feature can be balanced.

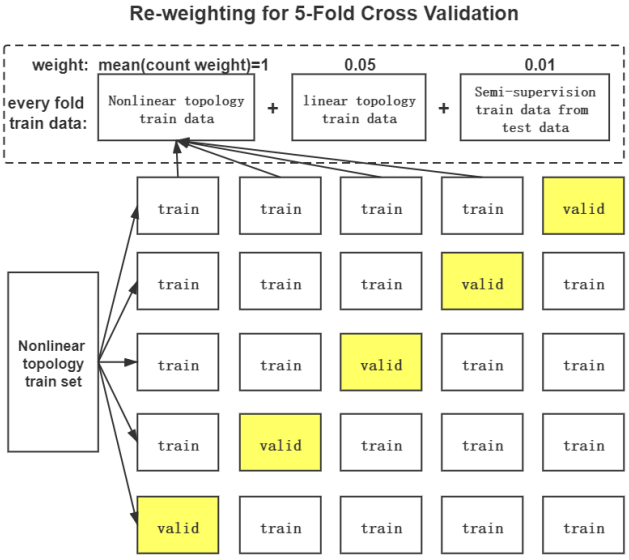

Figure 2: The Re-weighting method.

3.4. Semi-Supervised Training

In order to use the unlabeled data, a semi-supervised search method is proposed as shown by Alg.1, which randomly searches the available semi-supervised data from the test set. According to the 5-fold cross validation, 10 samples are selected from the test set to join each fold of the train set each time. As long as the cross validation score rises, they are retained, and the prediction of the test set which is also the label of semi-supervised data is updated at the same time.

Algorithm 1: Semi-Supervised Search

1: **Input:** The set of architecture-accuracy pairs as data D_0. Through 5-fold cross validation splitting D_0 to get the set of architecture-accuracy pairs D_1 as train data, and the set of architecture-accuracy pairs D_2 as valid data. Number of optimization iterations L. An empty architecture-accuracy pairs set D_4.

2: Generate N architectures as unlabeled architectures to use as same as test data.

3: Predict the N architectures to obtain the accuracy and form the labeled dataset D_3.

4: Use 5-fold cross validation of D_0 to get $score_0$.

5: **For** l = 1, · · · , L **do**

6: Randomly select 10 samples $\widetilde{D}$ from D_3 to join D_4.

7: For every fold, Set $D_1 \cup D_4$ as train data to get the $score_1$.

8: If $score_1 > score_0$:

$$score0 = score1,$$

$use\ the\ D_1 \cup D_4\ as\ train\ data\ \ to\ update\ the\ label\ of\ D_3$

9: Else:$del\ \widetilde{D}\ from\ D_4$

end for

Output: The set of architecture-accuracy pairs D_4 which can join the D_1 through 5-fold as the new train data.

After the search, the semi-supervised data set is added into five models of 5-fold. Only the train data set is added rather than the valid data set for avoiding the model over fitting its own prediction.

3.5. Model Deviation Correction

Model deviation correction method aims to train a modify model to correct the original predictor. Through 5-fold cross validation, we can get the 5-fold prediction of the valid set. The difference between the above prediction value and the original label value is taken as the new label of the data set, then we can train a modify model to get the modified prediction of test set. So that the process of correction is the original prediction of test set minus the modified prediction. The re-weighting is also used but the semi-supervised data set is not joined. After modifying, the prediction is more accurate.

4. Experiments

4.1. Training and Testing Details

In the experiment, the seed of cross validation split and the parameters of LightGBM are frozen. The metric of

LightGBM for early stop is RMSE, which is also the evaluation metric used in the competition. The L2 loss function is used, learning rate is 0.01, and the number of early stop rounds is 200.The simple NN model is composed of multi-layer full connection, BN, dropout and PRELU. The optimizer is Adam with learning rate 0.001, and batch size is set to the number of full data. After the semi-supervised training, the prediction of test data can be updated as label of semi-supervised data, iteration train and predict process three times to get more accurate semi-supervised label.

4.2. The value of Re-Weighting

With the original LightGBM model, the experiments only use the re-weighting method, and set different pow parameter and different weight of train set of linear topological network architecture. The experiment results are as follow.

Table 1: Comparisons on different pow parameter (The weight of train set of linear topological network is 0.1).

RMSE	0.2269	0.2134	**0.2068**	0.2085	0.2117
pow	**7**	6	**5**	4	3

Table 2: Comparisons on different weight of train set of linear topological network architecture (The pow parameter is 5).

RMSE	0.2026	0.1968	**0.1947**	0.1973	0.1992
weight	0.07	0.06	**0.05**	0.04	0.03

The best parameter configuration is found through experiments: when the pow is 5, the weight of train set of linear topological network architecture is 0.05.

4.3. Model Deviation Correction with Different Model

Use model correction for different models. In the first three columns, the original model and the modify model use the same type, while in the last two columns, the original model uses LightGBM and the modify model uses another one.

Table 3: Comparisons on different model deviation correction.

Model	LGB	XGB	CTB	LGB+XGB	**LGB+CTB**
RMSE	0.2291-	0.2364-	0.2305-	0.2291-	**0.2291-**
(-minus)	0.0047	0.0061	0.0028	0.0052	**0.0066**

The model deviation correction has effect on three types of tree models, the effect of using other modified model is better, which may due to the increase of diversity.

4.4. Component Ablation

In this section, based on the LightGBM model, we conducted ablation experiments of the three component, and the same common parameters were used in different experiments.

Table 4: Component Ablation experiments(re-weighting-RW, semi-supervised training-SST, model deviation correction-MDC)

Method	RW	SST	MDC	RMSE
LGB				0.2191
LGB+RW	√			0.1947
LGB+RW+SST	√	√		0.1805
LGB+RW+SST+MDC	√	√	√	**0.1755**

All three components can improve the performance of the model, and re-weighting has the most significant effect.

4.5. Semi-Supervised Training For NN Model

The RMSE of the cross validation of initial NN model is 0.2301. After adding the method designed in the paper, the score floats above 0.193. The SemiLGB's method can extend to different types of models. The LightGBM is better than NN, we think that the tree model has stronger generalization ability in the scenario absence of data.

5. Conclusion

SemiLGB is a semi-supervised learning algorithm. It uses a small set of high-quality architecture-accuracy pairs with different types to train the initial accuracy predictor, and then uses unlabeled architecture to further improve the predictor. The feature engineering can strengthen the important features. The re-weighting can utilize different types of network architecture and balance the long tail distribution of features, which is suitable for NAS to obtain architecture-accuracy pairs at different stages; The semi-supervised training can make use of unlabeled data; In addition, the model deviation correction enhances the performance of the model. The SemiLGB can be used in tree model, NN model and other models with various types to enhance the model performance in few-shot scenario.

References

[1] Renqian Luo, Xu Tan, et al. Semi-Supervised Neural Architecture Search. arXiv preprint arXiv:2002.10389, 2020.

[2] Yi Ren, Yangjun Ruan, Xu Tan, et al. Fastspeech: Fast, robust and controllable text to speech. In Advances in Neural Information Processing Systems, pages 3165–3174, 2019.

[3] Gabriel Bender, Pieter-Jan Kindermans, et al. Understanding and simplifying one-shot architecture search. In International Conference on Machine Learning, pages 549–558, 2018.

[4] Hieu Pham, Melody Guan, et al. Efficient neural architecture search via parameter sharing. In International Conference on Machine Learning, pages 4092– 4101, 2018.

[5] Hanxiao Liu, Karen Simonyan, et al. Darts: Differentiable architecture search. arXiv preprint arXiv:1806.09055, 2018.

[6] Ke, Guolin and Meng, Qi and Finley, Thomas and Wang, et al. LightGBM: A highly efficient gradient boosting decision tree. Conference and workshop on Neural Information Processing Systems, 30, 2017.