Performance Predict Method Based on Neural Architecture Search

Meili Zhou[1], Zongwen Bai[1], Tingting Yi[1], Xiaohuan Chen[1], Wei Wei[2]
[1] School of Physics and Electronic Information, Yan'an University, China
[2] School of Computer Science and Engineering, Xi'an University of Technology, China
zml@yau.edu.cn, whiteboy1999@sina.com, tt_y3658@163.com, GRdh1227@163.com, taneo@126.com

Abstract

Deep learning has granted remarkable breakthroughs on various tasks over the past few years, such as image segmentation, speech recognition, and nature language processing. One vital aspect of progress is the emergence of advanced neural architectures. However, Currently used architectures have frequently been developed manually by human experts, which is a time-consuming and laborious process. Because of this, more and more research is now involved in automated neural architecture search techniques.

This paper studies the process of Neural Architecture Search (NAS) technology, summarizes the previous work in this research field and classifies it according to three aspects: search space, search strategy, and acceleration method. In addition, this study selects the performance prediction method in the NAS acceleration strategy as a breakthrough direction and inherits the works of MetaQNN network structure and the sequential regression prediction model (SRMs), which were proposed by the previous research. Firstly, based on our hypothesis, we successfully use the idea of the N-grams model of natural language processing to extract the sequence features belonging to the chain neural network. Then, based on the extracted network structure features, referring to the steps of SRMs, we give a new recipe for predicting the accuracy score of neural network models on the training set. Finally, through experiments and comparison, we prove the accuracy of this prediction model and the effectiveness of accelerating the neural architecture search process.

Keywords: Neural architecture search, Sequence regression models, Performance prediction, Network structure feature

1 Introduction

The renaissance of Deep Learning Neural Networks (DNNs) has both created an active community of research in artificial intelligence and achieved many state-of-the-arts on intelligence processing system and significantly improved the performance on many works, for instance, image recognition [1-3], speech recognition [4], visual question answering [5-7], image or graph classification [8-9] and so on.

Despite recent breakthroughs in the theory of DNNs, for a sustained success and approaching perspectives of DNNs, inventing an efficient and powerful neural architecture requires extraordinary human effort and takes a long time is still a challenge. The other challenge is how to design a lightweight network for a special task.

Recently, Neural Architecture Search (NAS) has much attracted the interest of scientists [9-15] to deal with the above issues. The focus of NAS is the idea of using a search algorithm to get the architecture structure for the issue that we care about. The fundamental strategy is composing neural network architectures automatically, alternative relying heavily on expert experience and knowledge.

In this work, we propose the performance predict technique based on network architecture for NAS. It is a method providing guidance for NAS.

2 Relate Work

The target of neural network compression and network architecture search is to design a smaller and more efficient network oriented on the practical application requirement. The classical neural network compression method includes network pruning, low-rank decomposition, knowledge distillation, parameter sharing, and quantization. On the other hand, the network architecture search is easier to implement based on end to end network.

Network pruning [16-22]. It includes removing some unimportant weights from the model to produce a sparse weight matrix, or directly removing the whole matrix corresponding to the attention head to realize pruning of the model, and some models realize pruning through regularization.

Low-rank decomposition [23-26]. That is, the original large weight matrix decomposes multiple low-rank small matrices to reduce the amount of computation. This method can be used not only in a

*Corresponding Author: Zongwen Bai; E-mail: whiteboy1999@sina.com
DOI: 10.3966/160792642020032102007

label embedding to save disk memory but also in the parameter matrix of the feedforward layer or self-attention layer to speed up model training.

Knowledge distillation [27-31]. Through the introduction of teacher network to motivate the training of students' networks, knowledge transfer is realized. The teacher network has a complex structure to train the superior probability distribution, which is distilling the essence of the probability distribution from the complex structure, and then instructing the streamlined student network training, so as to realize the model compression, that is, knowledge distillation. In addition, distilling different network structures such as LSTMs from the Bert model, and further mining the network structure of teachers are expected to realize the continuous optimization of knowledge distillation.

Quantization [32-34]. By decreasing the number of bits desired for each parameter to compress the original network, memory can be significantly reduced.

Parameter sharing. Both the full connection layer and the self-attention layer realize parameter sharing, that is, all parameters in the encoder are shared, which not only reduces the number of parameters but also improves the training speed

In the past year, many deep learning researchers and practitioners have spent a lot of their time considering what kind of architecture of a neural network is profitable for their particular issue. And network architecture search is a cornerstone when deep learning research is application-oriented. The vast majority of current works on NAS.

Liu et al. [35] proposed a straightforward conversion of DARTS; for those not familiar, DARTS is one of stochastic optimization (gradient-based) way which initializes all potential architectures at the beginning and optimizes not only the parameters weights but also scalar weights on each path.

Dong et al. [36] incorporate the best of both worlds from ENAS and DARTS. DARTS already quoted as before; ENAS [37] is an RL-based method proposed by Pham et al. in which a giant graph is also initialized in the beginning analogous to DARTS, rather than additional variables on edges, the RL-based controller chooses which path is to be activated.

Baker et al. [10] regard one of the remarkable applications of NAS – search for architectures suit for fast inference on mobile apparatuses. In order to obtain that, they introduced a multi-objective optimization, where the RL-based controller is compelled to output an architecture not only with a super performance but also with low latency as a test on the CPU-core of Google Pixel 1.

It is thrilly and liberating to discover a large volume of papers overcoming the requirements of the amount of GPUs / TPUs you have, and achieving significant results. It would be doubtless interesting to find how NAS will improve and what other recipes researchers will propose. [19, 33-34, 38].

3 Methodology

MetaQNN [10] is a meta-modeling way depended on reinforcement learning to automatically produced high-performing CNN architectures for a special learning task. And the network feature extraction is via N-grams

Inspired by DeepArchitect [17] method, we take the N-grams method to extract MetaQNN network feature, and N-grams widely used in natural language processing according to the following step:

(1) The obtained neural network composed of the chain structure and network structure character description as input.

(2) Split the description character of the network structure, and extract the sequence of character which denote each layer's type.

(3) Construct feature vector: setting appropriate range of parameter N, where is the length of the gram, taking N-grams algorithm to deal with all the network character that obtained from the last step, and store the feature in some order.

(4) Compute the feature matrix: according to the stored vector, adapt N-grams to split rearrangement network successive again. If the N is consistent with the parameter that

(5) Construct feature vector, then feature vector as row, and the matrix which takes network architecture as column corresponding position is setting to 1, and then we will get the feature matrix of network architecture.

Table 1. The example feature matrix on SVHN dataset according above step

N	Size of feature matrix	Sparsity of eigenmatrix
3	(1000, 24)	0.999903026
5	(1000, 94)	0.999903026
8	(1000, 344)	0.999903026
10	(1000, 621)	0.999903026
13	(1000, 1113)	0.999903026
15	(1000, 1417)	0.999903026
18	(1000, 1681)	0.999903026

Table 1 show that the parameter N directly affects the dimension of eigenvector and feature matrix size, and the bigger N, the more sparsity of feature matrix.

Performance predict method of sequence regression models (SRMs) is one method that based on the learning curve. The aim is to describe the validation performance y_T of the neural network configuration $X \in \chi \subset \mathbb{R}^d$ at epoch $T \in \mathbb{Z}^+$ using prior performance observation $y(t)$, for every configuration X trained on T epochs, here record a time series $y(T) = y_1, y_2 ..... y_T$ of validation performance. we train a population of n configurations, getting a set as :

$$S = \{(X^1, y^1(T)), (X^2, y^2(T)).....(X^n, y^n(T))\}$$

This description formulation is originated rom [41], here extremely all architectures and hyperparameters search algorithms naturally collect S.

We also adopt a set features u_X as above mentioned, originated from the neural network configuration X, according to a subset of time-series performance $y(\tau) = (y_t)_{t=1,2...\tau}$ ($1 \le \tau \le T$) from S to train a regression models, where each successive model takes one more point of the time-series validation data. Compared with the method of training a single model in the following chapters, SRMs is more efficient and accurate in computation.

4 Experiment

4.1 Datasets

For the sake of contrast the performance on different datasets, we take the classical image datasets and natural language processing datasets. They are TinyImageNet [39], PTB [40], CIFA-10 [41] and SVHN respective and other applications [42-47].

The Tiny ImageNet dataset is a modified subset of the original ImageNet dataset [1]. Here, there are 200 different classes instead of 1000 classes of ImageNet dataset, with 10,000 validation examples and100,000 training examples. The resolution of the images is just 64x64 pixels, which makes it more challenging to extract information from it. A glance at the images shows that it is hard for the human detect objects in some images by eye.

The Penn Treebank (PTB) is dataset which project selected 2,499 stories from a three year Wall Street Journal (WSJ) and collection of 98,732 stories for syntactic annotation. It contain 2,499 stories in the Treebank. It consists of the raw text for each story. And three "map" compressed files (pennTB_tipster_ wsj_map.tar.gz) are provide for users who have licensed Treebank and give the relation between the 2,499 PTB filenames.

CIFAR-10 dataset is a elementary dataset for visual model trainning, it include 60000 colour images in 10 classes, that is to say, 6000 images per class, and each image pixes is 32x32. The number of training and test images is 50000 and 1000 respectively. Forthermore, the dataset is can be divide into five training batches and one test batch, each contain 10000 images. The training batches includes the remaining images with random order. The test batch contains exactly 1000 randomly-selected images from each class. but some training batches may contain more images from one class than another. Between them, the training batches contain exactly 5000 images from each class.

SVHN is a real-world image dataset oriented for exploiting machine learning or object recognition applications with minimal requirement on data formatting and preprocessing. It can be view as similar in flavor to MNIST (e.g., the images are of small cropped digits), but integrates an order of magnitude more labeled data (exceed 600,000 digit images) and originate from a unsolved, markedly harder, real world problem SVHN is collected from house numbers in Google Street View images. The dataset include 73257 training digits, 26032 digits for testing, additional, somewhat less difficult samples, to act as extra training data.

4.2 Different Network Architectures

We select three typical networks on different datasets to validate our schema.

We firstly select 500 ResNet architectures, and then train the model on the TinyImageNet dataset (which containing 200 classes with 500 training images and the size is 32 × 32) for 140 epochs. We alter filter sizes, depths and number of convolutional filter block outputs. And the filter size is chosse from {3, 5, 7}, and the number of filter is sample from {2, 3, 4, …, 22}, and each ResNet block is composed of three convolution layers, and batch normalization and summation layer sequentially. The number of blocks will shifting from 2 to 18. Network depth varies from 14 to 110. Every network trained 140 epochs via Nesterov optimizer, the learning rate is 0.1, decay rate is 0.1 and momentum is 0.9.

LSTM is suitable for processing sequence data and we choose 300 LSTM models to reflect the performance of our algorithm, and train the LSTM model on the Penn Treebank dataset via 60 epochs, and assessing perplexity on the validation set. We alter number of LSTM cells and hidden layer inputs between 10 and 1400. The step size is 20, each network trained 60 epochs, batch size is 50, and adopt random gradient descent to train the network. For the sake of avoid over-fitting matter, the dropout is 0.5, and the dictionary size is 400 to embed when dataset vectorization.

For comparison, we select 1,000 model architectures from the search space detailed by Baker et al. (2017) randomly, which addmits for altering the numbers and orderings of pooling, convolution, and fully connected layers. For the SVHN experiment, the models are between 1 and 12 layers, and for the CIFAR-10 experiment the models are between 1 and 18 layers. and all the architectures is trained on SVHN and CIFAR-10 datasets via 20 epochs.

In order to compare the relation between regression model and performance, we chose ordinary least squares (OLS), Bayesian linear regression (BLR), random forests (RF), and ν-support vector machine regression (ν-SVR) as regression and take 25% epochs of very 100 models.

Table 2. Model compare and selection

Model	dataset	v-SVR	Random Forest	OLS
MetaQNN	CIFAR-10	94.22 ± 0.25	92.27 ± 0.91	93.22 ± 1.1
ResNet	TinyImageNet	85.78 ± 1.82	91.37 ± 2.18	90.15 ± 1.8
LSTM	Penn Treebank	83.29 ± 7.71	91.38 ± 1.97	89.8 ± 0.16

Table 2 shows that the v-SVR have the best performace, and then, we will selecet this method to do the next procedure.

In oder to show the different performance on vary features, we chose features depended on architecture parameters (AP), time-series (TS), and hyperparameters (HP) validation performances.

Where AP include total the parameter of weights number and layers number. HP contain all hyperparameters that is used for training the neural networks. For instance, the initialized learning rate and learning rate decay. TS features expressed by the validation performances $y(\tau)=(y_t)_{t=1,2,...t}$. here we have trained the τ^{th} model in the SRM. the experement show as following:

Table 3. Different feature performance on various architecture

Feature Set	MetaQNN	ResNet	LSTM
TS	93.98 ± 0.15	86.52 ± 1.85	97.81 ± 2.45
AP	27.45 ± 4.25	84.33 ± 1.70	16.11 ± 1.13
HP	12.60 ± 1.70	8.78 ± 1.14	3.98 ± 0.88
TS+AP	84.09 ± 1.40	88.82 ± 2.95	96.92 ± 2.80
AP+HP	27.01 ± 2.50	81.71 ± 3.90	15.97 ± 2.57
TS+AP+HP	94.44 ± 0.14	91.8+1.10	98.24 ± 2.11

Table 3 indicate that TS feature is the most important factor for performance predication, It will show significant value in subsequent exploration.

Based on the above analysis, we can draw a conclusion that the main important feature is time series among the network architecture, hyperparameter and time series validation accuracy, but we believe that The performance of a network architecture is closely related to the network architecture itself, and we hope that predict the validation accuracy by machine learning way which depend on network architecture itself, and the most time-consuming process of model search is greatly reduced by obtaining the validation accuracy of some epochs without retraining the model. The above experiment result shown that the final performance of networks with similar sequence structure is also similar, this will be the basis of our subsequent experiments. for the extraction of the network themself feature, we take N-grams to extraction as description before. It is important to note that the feature extraction way is using chain network architecture sequence, namely, the type and sequence of each layer and the learning method is take the SRMs. Furthermore, we utilize sequential iteration to improve the predict accuracy [45-47].

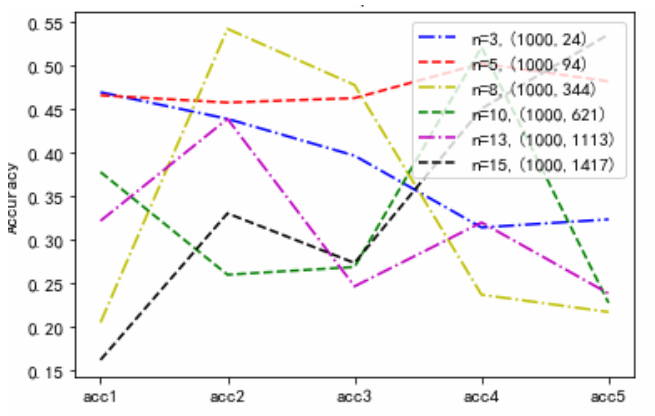
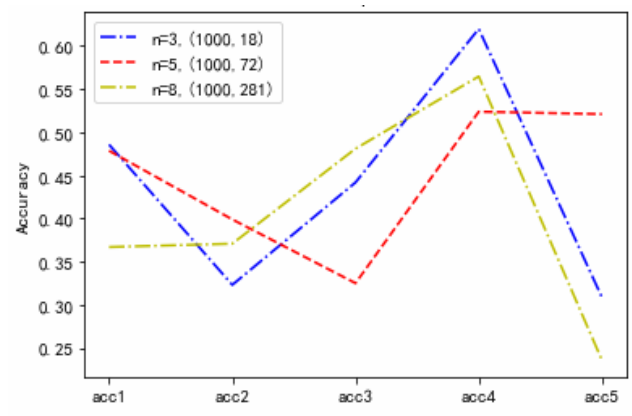
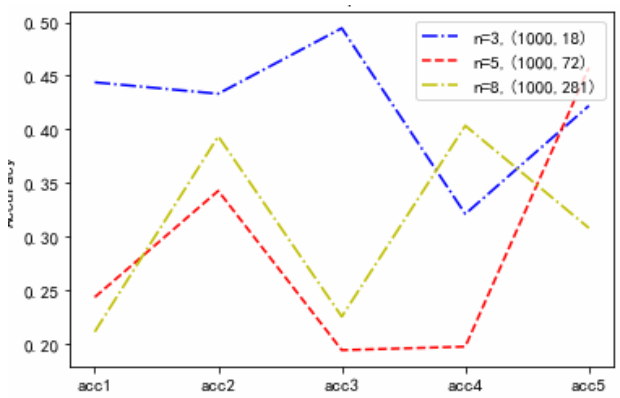


Figure 1. Random validation accuracy on SVHN

Figure 1 show that the varying of n in the process of feature extraction by N-grams, the network is MetaQNN and training dataset is SVHN. The different type denote the parameter n and corresponding size of feature matrix.



(a) Random validation accuracy by five times sample without normalization



(b) Random validation accuracy by five times sample with normalization.

Figure 2. Compare the random validation accuracy

Consider the feature matrix high sparsity and large difference in feature value, we utilize normalization way to offset these effects, the result show as Figure 2. When adjust the parameter n, the predict accuracy will

be changing, but it does not indicate that the bigger n the better performance, and too large n will decrease the predict accuracy.

Figure 2 show that normalization will promote the variance diminish when take same parameter and dataset. Figure 2(a) is the random validation accuracy by five times sample without normalization and Figure 2(b) is the random validation accuracy by five times sample with normalization.

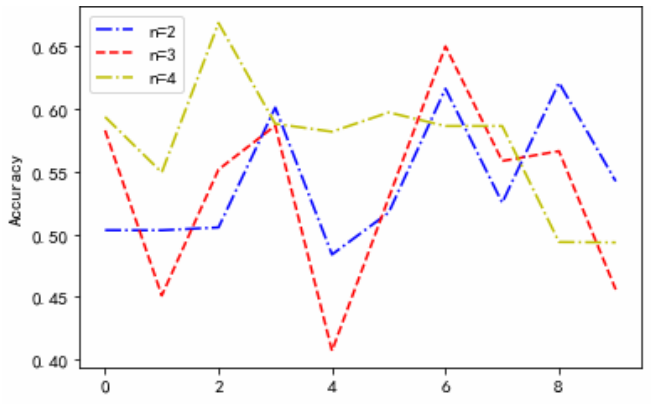


Figure 3. Random validation accuracy on CIFAR-10

Figure 3 shows the performance of random validation accuracy on CIFAR-10 dataset under ten times sample, here n = 2, 3, 4.

Figure 4 compare the random validation performance on CIFAR-10, it is clear that compare with the random predicate just as figure1 and figure 2 shows, add one or two epochs is promote predicate accuracy notable

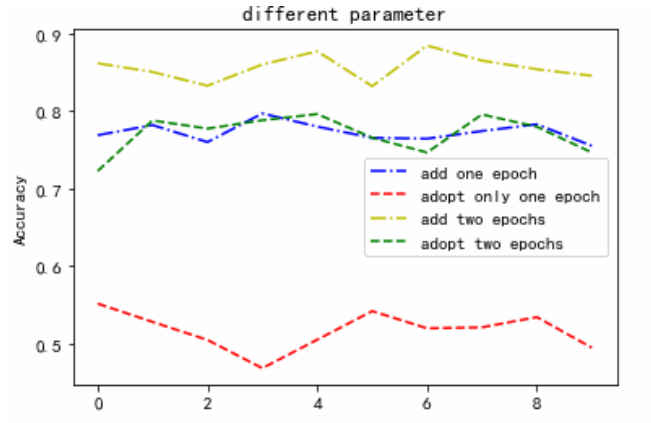


Figure 4. Random validation accuracy on CIFAR-10

5 Conclusion

From above experiment we can get the following conclusion to facilitate the process of NAS:

Using SRMs predicate network architecture by network structure description as feature extraction, the performance approximate 40% as take MetaQNN network training on SVHN dataset, Comparatively speaking, the performance will close to 50%-60% when take the same network on CIFAR-10 dataset, and have stable performance. It indicate that the model depend on datasets.

For the MetaQnn network training on CIFAR-10, the predicate performance will close to 72% by add one epoch with same predicate model, However, if only use one epoch to predicate the performance, the accuracy is only 50%. Further exploration, using two epochs and add two epochs within same condition, predicate accuracy increased from 75% to 92%. all the result illustrate that increase the training epochs is the better way to make better performance.

Acknowledgements

I would like to thank all reviewers of this paper for their hard work, which motivated me a lot. I would like to thank the anonymous referee that provided the preliminary study.

This work was supported by the National Natural Science Foundation of China (Grant No. 61941112, 61761042, 61763046), Key Research and Development Program of Yanan (Grant No. 2017KG-01, 2017WZZ-04-01). This job is supported by the Key Research and Development Program of Shaanxi Province (No. 2018ZDXM-GY-036), This job is also supported by the Key Research and Development Program of Yanan university (No. CXY201909, YDZ2019-05) and Shaanxi Key Laboratory of Intelligent Processing for Big Energy Data (No. IPBED7, IPBED10).

References

[1] K. He, X. Zhang, S. Ren, J. Sun, Deep Residual Learning for Image Recognition, *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Las Vegas, NV, USA, 2016, pp. 770-778.

[2] A. Krizhevsky, I. Sutskever, G. E. Hinton, Imagenet Classification with Deep Convolutional Neural Networks, *25th International Conference on Neural Information Processing Systems (NIPS)*, 2012, Lake Tahoe, Nevada, USA, pp.1097-1105.

[3] B. Fang, Y. Li, H. Zhang, J. C.-W. Chan, Hyperspectral Images Classification Based on Dense Convolutional Networks with Spectral-Wise Attention Mechanism, *Remote Sensing*, Vol. 11, No. 2, Article 159, January, 2019.

[4] G. Hinton, L. Deng, D. Yu, G. E. Dahl, A.-r. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. N. Sainath, B. Kingsbury, Deep Neural Networks for Acoustic Modeling in Speech Recognition: The Shared Views of Four Research Groups, *IEEE Signal Processing Magazine*, Vol. 29, No. 6, pp. 82-97, November, 2012.

[5] P. Wang, Q. Wu, C. Shen, A. Dick, A. van den Hengel, Fvqa: Fact-based Visual Question Answering, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 40, No. 10, pp. 2413-2427, October, 2018.

[6] Q. Wu, C. Shen, P. Wang, A. Dick, A. van den Hengel, Image

Captioning and Visual Question Answering Based on Attributes and External Knowledge, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 40, No. 6, pp. 1367-1381, June, 2018.

[7] P. Wang, Q. Wu, C. Shen, A. van den Hengel, The VQA-machine: Learning How to Use Existing Vision Algorithms to Answer New Questions, *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Honolulu, HI, USA, 2017, pp. 3909-3918.

[8] D. Wang, Y. Li, L. Ma, Z. Bai, J. C.-W. Chan, Going Deeper with Densely Connected Convolutional Neural Networks for Multispectral Pansharpening, *Remote Sensing*, Vol. 11, No. 22, Article 2608, November, 2019.

[9] Z. Bai, J. Tu, Y. Shi, An Improved Algorithm for the vertex Cover P3 Problem on Graphs of Bounded Treewidth, *Discrete Mathematics & Theoretical Computer Science*, Vol. 21, No. 4, Article 17, November, 2019.

[10] B. Baker, O. Gupta, N. Naik, R. Raskar, Designing Neural Network Architectures Using Reinforcement Learning, *International Conference on Learning Representation (ICLR)*, Toulon, France, 2017, pp. 1-18.

[11] L. Chen, M. Collins, Y. Zhu, G. Papandreou, B. Zoph, F. Schroff, H. Adam, J. Shlens, Searching for Efficient Multi-scale Architectures for Dense Image Prediction, *Advances in Neural Information Processing Systems (NIP)*, Montreal, Canada, 2018, pp. 8713-8724.

[12] K. Kandasamy, W. Neiswanger, J. Schneider, B. Poczos, E. Xing, Neural Architecture Search with Bayesian Optimisation and Optimal Transport, *Advances in Neural Information Processing Systems (NIPS)*, Montreal, Canada, 2018, pp. 2020-2029.

[13] X. Li, Y. Zhou, Z. Pan, J. Feng, Partial Order Pruning: For Best Speed/Accuracy Trade-off in Neural Architecture Search, *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Long Beach, CA, USA, 2019, pp. 9137-9145.

[14] R. Luo, F. Tian, T. Qin, E. Chen, T. Liu, Neural Architecture Optimization, *Advances in Neural Information Processing Systems (NeurIPS)*, Montreal, Canada, 2018, pp.7827-7838.

[15] W. Wei, Z. Sun, H. Song, H. Wang, X. Fan, X. Chen, Energy Balance-based Steerable Arguments Coverage Method in WSNs, *IEEE Access*, Vol. 6, pp. 33766-33773, March, 2017.

[16] J. Yoon, S. Hwang, Combined Group and Exclusive Sparsity for Deep Neural Networks, I *34th International Conference on Machine Learning (ICML 2017)*, Sydney, NSW, Australia, 2017, pp. 3958-3966.

[17] J. Luo, J. Wu, W. Lin, Thinet: A Filter Level Pruning Method for Deep Neural Network Compression, *IEEE International Conference on Computer Vision*, Venice, Italy, 2017, pp. 5068-5076.

[18] M. Ren, A. Pokrovsky, B. Yang, R. Urtasun, Sbnet: Sparse Blocks Network for Fast Inference, *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Salt Lake City, UT, USA, 2018, pp. 8711-8720.

[19] Z. Liu, J. Li, Z. Shen, G. Huang, S. Yan, C. Zhang, Learning Efficient Convolutional Networks through Network Slimming, *IEEE International Conference on Computer Vision*, Venice, Italy, 2017, pp. 2755-2763.

[20] Y. He, X. Zhang, J. Sun, Channel Pruning for Accelerating Very Deep Neural Networks, *IEEE International Conference on Computer Vision*, Venice, Italy, 2017, pp. 1398-1406.

[21]X. Sun, X. Ren, S. Ma, H. Wang, Meprop: Sparsified Back Propagation for Accelerated Deep Learning with Reduced Overfitting, *34th International Conference on Machine Learning*, Sydney, NSW, Australia, 2017, pp. 3299-3308.

[22] D. Molchanov, A. Ashukha, D. Vetrov, Variational Dropout Sparsifies Deep Neural Networks, *34th International Conference on Machine Learning*, Sydney, NSW, Australia, 2017, pp. 2498-2507.

[23] W. Wang, Y. Sun, B. Eriksson, W. Wang, V. Aggarwal, Wide Compression: Tensor Ring Nets, *IEEE Conference on Computer Vision and Pattern Recognition*, Salt Lake City, UT, USA, 2018, pp. 9329-9338.

[24] J. Yim, D. Joo, J. Bae, J. Kim, A Gift from Knowledge Distillation: Fast Optimization, Network Minimization and Transfer Learning, *IEEE Conference on Computer Vision and Pattern Recognition*, Honolulu, HI, USA, 2017, pp. 7130-7138.

[25] I. Oseledets, Tensor-train Decomposition, *SIAM Journal on Scientific Computing*, Vol. 33, No. 5, pp. 2295-2317, September, 2011.

[26] Y. Kim, E. Park, S. Yoo, T. Choi, L. Yang, D. Shin, *Compression of Deep Convolutional Neural Networks for Fast and Low Power Mobile Applications*, https://arxiv.org/abs/1511.06530.

[27] S. Zagoruyko, N. Komodakis, P*aying More Attention to Attention: Improving the Performance of Convolutional Neural Networks Via Attention Transfer*, https://arxiv.org/abs/1612.03928.

[28] G. Chen, W. Choi, X. Yu, T. Han, M. Chandraker, Learning Efficient Object Detection Models with Knowledge Distillation, *Advances in Neural Information Processing Systems*, Long Beach, CA, USA, 2017, pp. 742-751.

[29] Z. Lu, V. Sindhwani, T. Sainath, Learning Compact Recurrent Neural Networks, *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Shanghai, China, 2016, pp. 5960-5964.

[30] C. Leng, Z. Dou, H. Li, S. Zhu, R. Jin, Extremely Low Bit Neural Network: Squeeze the Last Bit Out with Admm, *Thirty-Second AAAI Conference on Artificial Intelligence*, New Orleans, Louisiana, USA, 2018, pp. 3466-3473.

[31] Q. Hu, P. Wang, J. Cheng, From Hashing to Cnns: Training Binary Weight Networks Via Hashing, *Thirty-Second AAAI Conference on Artificial Intelligence*, New Orleans, Louisiana, USA, 2018, pp. 3247-3254.

[32] P. Wang, Q. Hu, Y. Zhang, C. Zhang, Y. Liu, J. Cheng, Two-step Quantization for Low-bit Neural Networks, *IEEE Conference on Computer Vision and Pattern Recognition*, Salt Lake City, UT, USA, 2018, pp. 4376-4384.

[33] U. Köster, T. Webb, X. Wang, M. Nassar, A. K. Bansal, W. Constable, O. Elibol, S. Hall, L. Hornof, A. Khosrowshahi, C. Kloss, R. Pai, N. Rao, Flexpoint: An Adaptive Numerical

Format for Efficient Training of Deep Neural Networks, *Advances in Neural Information Processing Systems*, Long Beach, CA, USA, 2017, pp. 1742-1752.

[34] D. Alistarh, D. Grubic, J. Li, R. Tomioka, M. Vojnovic, QSGD: Communication-Efficient SGD via Gradient Quantization and Encoding, In *Neural Information Processing Systems*, Long Beach, CA, USA, 2017, pp. 1709-1720.

[35] C. Liu, L-C. Chen, F. Schroff, H. Adam, W. Hua, A. Yuille, F. Li, Auto-DeepLab: Hierarchical Neural Architecture Search for Semantic Image Segmentation, *IEEE Conference on Computer Vision and Pattern Recognition*, Long Beach, CA, USA, 2019, pp. 82-92.

[36] X. Dong, Y. Yang, Searching for a Robust Neural Architecture in Four GPU Hours, *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Long Beach, CA, USA, 2019, pp. 1761-1770.

[37] H. Pham, M. Guan, B. Zoph, Q. Le, J. Dean, Efficient Neural Architecture Search via Parameter Sharing, *Proceedings of the 35th International Conference on Machine Learning*, Stockholm, Sweden, 2018, pp. 4092-4101.

[38] M. Tan, B. Chen, R. Pang, V. Vasudevan, M. Sandler, A. Howard, Q. V. Le, MnasNet: Platform-Aware Neural Architecture Search for Mobile, *The IEEE Conference on Computer Vision and Pattern Recognition*, Long Beach, CA, USA, 2019, pp. 2815-2823.

[39] L. Yao, J. Miller, Tiny Imagenet Classification with Convolutional Neural Networks, CS 231N, http://cs231n.stanford.edu/reports/2015/pdfs/leonyao_final.pdf, 2015.

[40] E. K. Ringger, R. C. Moore, E. Charniak, L. Vanderwende, H. Suzuki, Using the Penn Treebank to Evaluate Non-Treebank Parsers, *Proceedings of the Fourth International Conference on Language Resources and Evaluation (LREC'04)*, Lisbon, Portugal, 2004, pp. 867-870.

[41] A. Krizhevsky, *Learning Multiple Layers of Features from Tiny Images*, Technical Report TR-2009, University of Toronto, Toronto, Canada, April, https://www.cs.toronto.edu/~kriz/learning-features-2009-TR.pdf, 2009.

[42] Y. Netzer, T. Wang, A. Coates, A. Bissacco, B. Wu, A. Y. Ng, Reading Digits in Natural Images with Unsupervised Feature Learning, *NIPS Workshop on Deep Learning and Unsupervised Feature Learning*, Granada, Spain, 2011, pp. 1-9.

[43] W. Wei, Y. Qi, Information Potential Fields Navigation in Wireless Ad-Hoc Sensor Networks, *Sensors*, Vol. 11, No. 5, pp. 4794-4807, May, 2011.

[44] W. Wei, Y. Qiang, J. Zhang, A Bijection between Lattice-Valued Filters and Lattice-Valued Congruences in Residuated Lattices, *Mathematical Problems in Engineering*, Vol. 2013, Article ID 908623, July, 2013.

[45] W. Wei, X. Yang, B. Zhou, J. Feng, P. Shen, Combined Energy Minimization for Image Reconstruction from Few Views, *Mathematical Problems in Engineering*, Vol. 2012, Article ID 154630, October, 2012.

[46] Q. Ke, J. Zhang, H. Song, Y. Wan, Big Data Analytics Enabled by Feature Extraction Based on Partial Independence, *Neurocomputing*, Vol. 288, pp. 3-10, May, 2018.

[47] W. Wei, X. Fan, H. Song, X. Fan, J. Yang, Imperfect Information Dynamic Stackelberg Game based Resource Allocation Using Hidden Markov for Cloud Computing, *IEEE Transactions on Services Computing*, Vol. 11, No.1, pp. 78-89, January-February, 2018.

Biographies

Meili Zhou received the M.S. degree in signal and information processing from the yanan university in 2008. She is a associate Professor with the School of physics and electronic information, yanan University. Her Interests cover signal processing, computer vision and image processing.

Zongwen Bai received the MS degree in yanan university 2008. He is currently pursuing the Ph.D. degree with the School of Computer Science, Northwestern Polytechnical University, Xi'an. He is an associate professor with the School of physics and electronic information, Yanan University, His research interests cover computer vision, nature language processing and deep learning.

Tingting Yi received the B.E. from Baoji University of Arts and Sciences, Shaanxi, China. She has been a graduate student major in Communication and information System at Yan'an University. Her research interests include AI, machine learning and deep learning.

Xiaohuan Chen received the B.S degree from Baoji Universty, major: Electronic Information Engineering. Since 2019, she has been a graduate student in Singnal and Information Processing at Yan'an University. Her research interests include compuer vision and AI.

Wei Wei (SM'17) received the M.S. and Ph.D. degrees from Xi'an Jiaotong University, Xi'an, China, in 2005 and 2011, respectively. He is currently an Associate Professor with the School of Computer Science and Engineering, Xi'an University of Technology, Xi'an. His current research interests include the area of wireless networks, wireless sensor networks application, image processing, mobile computing. He has published around 100

research papers in international conferences and journals.