Design Considerations of Large-Scale RRAM-Based Convolutional Neural Networks with Transfer Learning

Preprint · July 2018

DOI: 10.13140/RG.2.2.18727.37281

6 authors, including:

Dong Zhen
University of California, Berkeley
55 PUBLICATIONS **3,511** CITATIONS
SEE PROFILE

Jinfeng Kang
Peking University
494 PUBLICATIONS **12,610** CITATIONS
SEE PROFILE

Design Considerations of Large-Scale RRAM-Based Convolutional Neural Networks with Transfer Learning

Z. Dong[1,2], H. Li[1], D. Zhu[2], P. Huang[2], J. F. Kang[2#], and H.-S. P. Wong[1*]
[1]Department of Electrical Engineering and SystemX Alliance, Stanford University, Stanford, CA 94305, USA
[2]Institute of Microelectronics, Peking University, Beijing 100871, China
E-mail: [#]kangjf@pku.edu.cn; [*]hspwong@stanford.edu

***Abstract*—Resistive random access memory (RRAM) array can enable massive acceleration of neural network algorithms through in-memory computing capabilities. Related works mainly focus on small networks while large-scale state-of-the-art networks are more in need of acceleration due to the intensive computing and memory accesses. In this work, we address the design considerations of large-scale RRAM-centric neural networks for the first time: 1) both small and large RRAM-based hardware networks are simulated based on multilevel as well as analog characteristics of RRAM; 2) the influence of key design aspects on the system-level performance, such as the variation of RRAM conductance and the quantization methods, are analyzed comparing small and large networks; 3) transfer learning is introduced to ease restrictions on RRAM characteristics in the case of large-scale neural networks; 4) two schemes to achieve transfer learning are developed, with detailed discussions of advantages and disadvantages. This work provides guidelines for the RRAM-based implementation of large-scale neural networks that can address ImageNet-level or transfer learning tasks.**

Keywords—resistive random access memory, transfer learning, convolutional neural networks, in-memory computing.

I. INTRODUCTION

Neural networks have succeeded in many fields, such as image classification, object detection, natural language processing, etc. [1]-[3] These learning-based models can be huge and are typically driven by abundant data. For example, VGG-Net has 138 million weights, while the ImageNet dataset has 1.2 million training images with 1000 categories. Under conventional Von Neumann architecture, the movement of those parameters between memory and computing units could be energy-hungry and time-consuming. [4]-[5]

Non-volatile memory, such as resistive random access memory (RRAM), phase change memory (PCM) and spin-transfer torque magnetic random access memory (STT-MRAM), can enable in-memory computing to overcome Von Neumann bottleneck and the nature of massive parallelism provides unique opportunities for efficient neural network acceleration. [6]-[9] In this work, we utilize RRAM for its low operating energy, high switching speed, high endurance, multilevel and analog characteristics, and the potential for 3D integration. [10]

Many works relating to RRAM-based small networks have been done. [11]-[17] Nevertheless, accelerating large-scale neural networks is more crucial since CPU and GPUs can handle small networks in real time because of fewer parameters, but need even days to train a state-of-the-art neural network.

In order to achieve RRAM-based hardware neural networks, some difficult problems are supposed to be tackled beforehand. First, we need to use RRAM conductance states to represent full-precision software weights. Mapping continuously distributed full-precision weights to several (like 3 or 4) discrete conductance states may cause significant decrease in the classification accuracy. Moreover, the variation of conductance states makes the RRAM-represented weights inaccurate and thus will adversely affect as well as cause fluctuation in the performance of hardware neural networks. Finally, high endurance of RRAM devices is also important if the training process has too many iterations.

In this work, we analyze the design considerations and device-level requirements of RRAM technology in large-scale convolutional neural networks and develop feasible schemes in order to achieve transfer learning. Transfer learning means using the knowledge learned in one database to achieve better learning results on the other database, which is naturally suitable for RRAM-based hardware implementation. In transfer learning, the former layers of large-scale neural networks are fixed and only last layers will be trained on the new database, which frees the network from multilayer backpropagation and relaxes the restriction on RRAM devices. All tested RRAM characteristics to be used in simulations are presented in Section II. Section III shows the results and detailed analyses of RRAM-based small or large networks. Two schemes and practicable methods to achieve transfer learning are explained in Section IV. Finally comes the conclusion in Section V.

II. RRAM CHARACTERISTICS

Fig.1 shows the typical current-voltage curve of our fabricated RRAM device. The black curve corresponds to set process while the red curve corresponds to reset process. Generally, low conductance state can be abruptly set to high conductive state, while the reset process from high conductance to low conductance tends to be gradual.

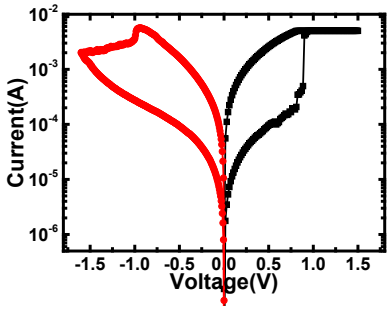

Fig.1 Typical DC I-V characteristics of a RRAM device.

This work is supported in part by the member companies of the Stanford Non-Volatile Memory Technology Research Initiative (NMTRI) affiliate program, and Systems on Nanoscale Information Fabrics (SONIC) Center, one of six centers of Semiconductor Technology Advanced Research Network (STARnet), a Semiconductor Research Corporation (SRC) program sponsored by Microelectronics Advanced Research Corporation (MARCO) and Defense Advanced Research Projects Agency (DARPA), and the NCN-NEEDS program, which is funded by the National Science Foundation, contract 1227020-EEC, and by the Semiconductor Research Corporation. This work is also supported in part by 973 Program (2011CBA00600) and NSFC Program (61334007). Z. Dong is supported in part by the Stanford UGVR Program during the summer of 2017.

By applying different AC pulses, with the peak values and pulse width adjusted, multilevel conductance states can be obtained. [18] As **Fig.2** presents, since the set process is abrupt, the high conductance state related to the black curve can be obtained by applying a 1.44V 100ns set pulse on every initial RRAM conductance states. Using the HCS as a starting point, low conductance states correspond to the red, orange, blue curve can be obtained by adding a -1.66V, -1.6V, -1.56V reset pulse respectively, with each 100ns pulse width.

The conductance proportion among those three low conductance states is around 1:3:8, and more conductance states can be reached by changing the reset pulses. In addition, the variations of those conductance states in **Fig.2** are approximately 27.4%, 14.5%, 10.7%, corresponds to LCS1, LCS2, LCS3 respectively. Generally, the variation of a low conductance state tends to be larger than that of a high conductance state. Since the high weight values actually play a more important role in the CNN system, the problem of relatively large variations of low weight values can be naturally settled. In addition, the RRAM tested in **Fig.2** can be easily reset to low conductance like 0.5 μ S, which can be utilized as 0 in the simulation of hardware neural networks.

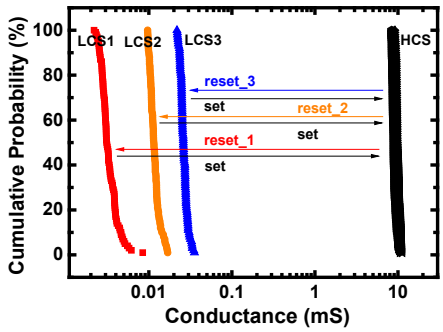

Fig.2 Multilevel conductance states which will be utilized as quantified weights in the hardware neural networks, and related operation methods for RRAM devices.

Fig.3 shows the analog characteristics of RRAM device. As **Fig.3 (a)** shows, up to hundreds of conductance states can be obtained by applying continuous small reset pulses on RRAM. [19] However, the operation to get analog characteristics contains hundreds of pulses and thus is time-consuming. As a result, analog RRAM can't be used to represent all weights in large-scale neural networks. Moreover, the change of RRAM conductance is nonlinear, which will have bad influence on the performance. We can use one stage instead of the whole curve of conductance change in order to get better linearity. Besides, based on the conductive filament theory of RRAM, the conductance change in this gradual reset process can be divided into three stages where the speed and range of conductance change are different. We have to consider the trade-off between large range and low speed when we use the analog characteristics, which will be discussed in Section IV.

Since only the reset process is gradual, the RRAM conductance, namely weight values, can't be increased during the training period. We can use the difference of two analog RRAM conductance as one weight value and thus can achieve both the potentiation and depression. Besides, we can also use the unbalanced set-reset pairs to obtain analog set process. As **Fig.3 (b)** shows, the RRAM device is firstly set by a 1.4V, 500ns pulse and then reset by a -1.44V, 5μs pulse in one iteration. Although designed unbalanced set-reset pairs can potentiate RRAM during each iteration, two time-consuming pulses are needed in this scheme. In addition, the stability of this scheme is worse than that of the analog characteristics shown in **Fig.3 (a)**.

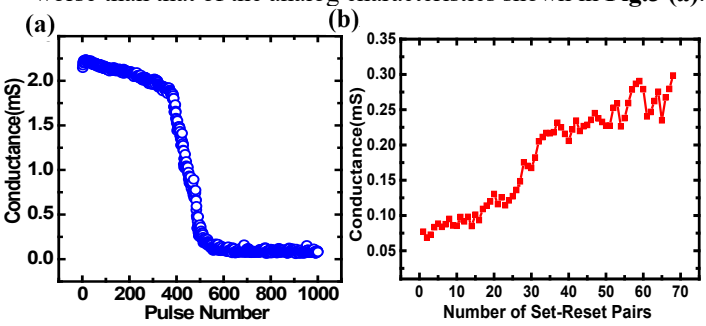

Fig.3 (a) Conductance states obtained by adding increasing number of small reset pulses. (b) Analog characteristics obtained by applying unbalanced set-reset pairs.

III. SIMULATION OF RRAM-BASED NUERAL NETWORKS

A. Small Networks on MNIST Database

MNIST handwritten database contains 60000 training images and 10000 test images. The size of each grayscale image is 28×28. The neural network architecture used for simulation consists of two layers: one convolutional layer and one fully-connected layer. After convolutional operation, the outputs of the first layer will then go through pooling and activation operations, and finally become inputs of the fully-connected layer. **Fig.4** shows the influence of kernel number and the number of alternative weight values on the recognition accuracy. [20] It can be pointed out that when the number of kernels is larger than 10, adding more convolutional kernels won't cause a significant increase in the recognition accuracy, while reducing the alternative weight values used for quantization won't result in bad performance too.

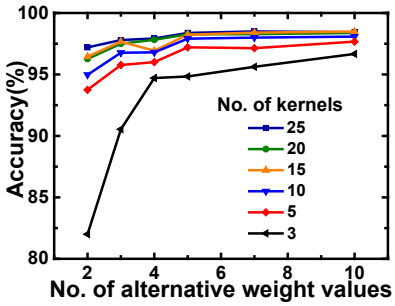

Fig.4 The influence of kernel number and the number of alternative weight values on the final performance of the CNN system.

Fig.5 illustrates the effect that the variation of conductance states has on the recognition accuracy. [20] When the variation level of conductance states is lower than 50%, there won't be considerable decrease in the recognition accuracy.

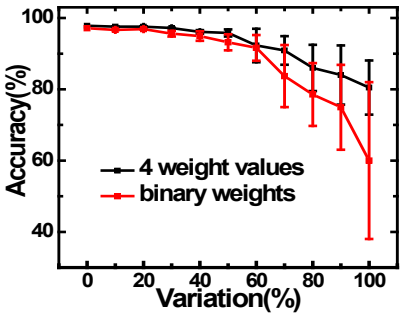

Fig.5 Recognition accuracy as a function of the variation level of RRAM conductance states.

Moreover, we can see from **Fig.5** that the fluctuation of the whole CNN system becomes larger as the variation of RRAM conductance gets bigger.

B. Large-scale Neural Netrorks on ImageNet Database

ImageNet large scale visual recognition challenge (ILSVRC) contains 1.2 million RGB images with size around 500*400 as training dataset, which takes about 138GB memory space. Besides, there is a test dataset consists of 0.1 million images and a validation dataset consists of 50,000 images. [21] Begin with AlexNet in 2012, a series of neural network benchmarks are proposed on the ImageNet database, such as VGG and GoogleNet in 2014 and ResNet in 2015. [22]-[25] We choose VGG-16 as the representation of traditional convolutional neural network architectures, which are formed by an ordered queue of convolutional layers, activation and pooling operations, and fully-connected layers. Specifically, VGG-16 contains 13 convolutional layers and 3 fully-connected layers. **Fig.6** shows the data flow through the whole VGG-16 network, where each cuboid represents the outputs of the former layer which are equivalent to the inputs of the latter layer. As for the representation of state-of-the-art architectures, we choose Google Inception version3 for its good performance and small parameter counts. Based on simulations, the results of Inception V3 are similar to those of the VGG-16, thus we will only introduce the results based on VGG-16 in this section.

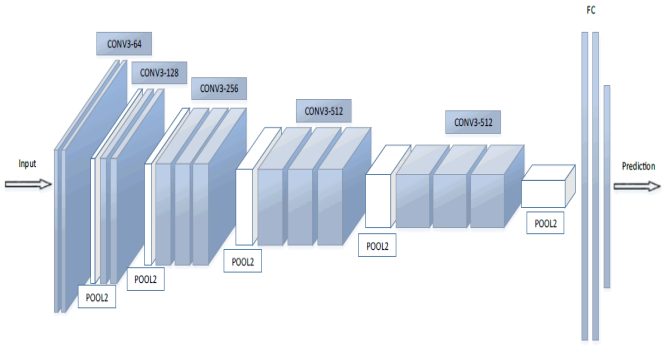

Fig.6 Illustration of the data flow in the VGG-16 neural network.

In order to achieve RRAM-based hardware implementation of VGG-16 network, a quantization scheme is needed to map full-precision weights to limited RRAM conductance states. Although there are schemes in the field of model compression, such as weight sharing or XNOR-Net methods, auxiliary circuits or additional memory are needed in the hardware implementation. [26]-[27] Since we have transfer learning process, exact imitation of the original network may not be necessary, thus we can directly map full-precision weights to the tested RRAM conductance states without pre-processing the model.

As **Fig.7** shows, in order to represent negative weight values in the RRAM crossbar array, the difference of two sum currents is used as one output value. Thus the equivalent weights are represented by the difference of two RRAM conductance states, namely $W_{eff} = W_{i,j,a} - W_{i,j,b}$. To take more advantages of using two RRAM as one weight, we can use multilevel RRAM whose conductance states are not equal to the difference among those states. For example, if the proportion of three discrete RRAM conductance states is 1:3:8, then the differences among those states are 2, 5, 7, which enables us to use 0, ±1, ±2, ±3, ±5, ±7, ±8 in the quantization scheme. It should be pointed out that we use the original high resistance state of RRAM device as 0 conductance. If the RRAM is not formed, then the original resistance can be extremely high. Even if the RRAM is a forming-free device, based on simulations, resistive window larger than 10 can prevent the system from considerable recognition accuracy decrease.

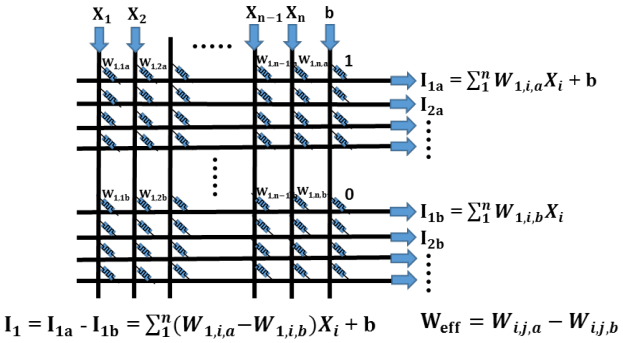

Fig.7 Schematic of using RRAM crossbar array for calculating the product of a vector (X) and a matrix (W).

Typically, the frequency histogram of weights in each layer approximately follows a normal distribution. For example, **Fig.8 (a)** shows the frequency histogram of weights in the twelfth convolutional layer, and **Fig.8 (b)** shows the histogram of weights in the first fully-connected layer. By multiplying a scaling factor, weights of all layers can share the same quantization circuit. Based on simulations, there are some weights with huge values (often 3 or 4 times larger than the standard deviation of weights) but small counts (too small to see in the frequency histogram) that play an important role in the convolutional neural network system. Thus choosing a small scaling factor and only focusing on values with large counts is not the best scheme. For example, the final top-5 classification accuracy of the quantization scheme with small scaling factor, which corresponds to the blue quantization results in **Fig.8 (a)** and **(b)**, is 12.3%. However, the top-5 classification accuracy of the quantization with optimized scaling factor, which corresponds to the black quantization results, is 50.4%. Although the effect of scaling factor will decline when the number of alternative weight values increases, selecting large conductance states to better represent those high-value weights can also benefit the performance of the large-scale CNN system when the variation level of conductance states becomes relatively high.

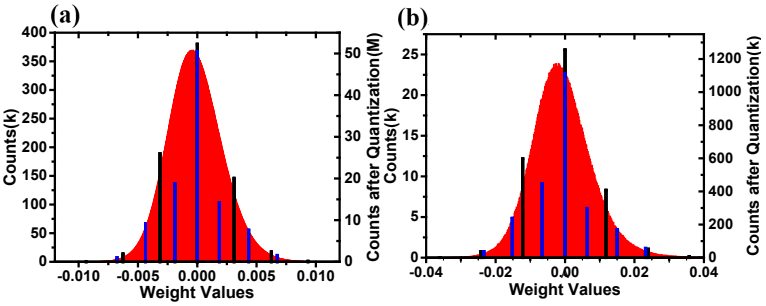

Fig.8 (a) Frequency histogram of weights in the first fully-connected layer. (b) Frequency histogram of weights in the twelfth convolutional layer (Conv5-2). The blue\black lines in (a) and (b) are the results after a quantization using small scaling factor\ appropriate scaling factor.

Fig.9 presents the classification accuracy as a function of the number of alternative weight values. For each input image, the VGG-16 network will generate a score vector as the output,

which contains 1000 values that correspond to 1000 categories of ImageNet. Top-5 accuracy can be obtained by following the protocol: if one of the top 5 scores in the output vector correspond to the right category, then the classification of this input image is considered successful. In order to analyze the restriction of RRAM characteristics in large-scale neural networks, some bad situations with severe non-ideal effects need to be simulated. Thus we add top-10 accuracy to better understand the trend when the classification accuracy is low. Without data augmentation which is difficult for hardware implementation, we train a VGG-16 network and get 69% top-1 accuracy, 89% top-5 accuracy and 93% top-10 accuracy on the validation dataset. Although these accuracies are a bit (like 3%) lower than those published in [23], they are definitely enough for us to analyze the trend. From **Fig.9** we can see that when the number of alternative weight values is larger than 10, which can be achieved by using 4 conductance states with proportion 1:3:8:12, there will be no considerable accuracy decrease of the VGG-16 system. Moreover, the influence of quantization on large-scale networks is much severer than that on small-scale networks. For example, if we use 6 alternative weight values (like the 1:3:8 scheme), zero accuracy loss can be achieved in small networks, while we have to face about 20% accuracy decrease in the VGG-16 network. (There might be slight variance due to different quantization schemes).

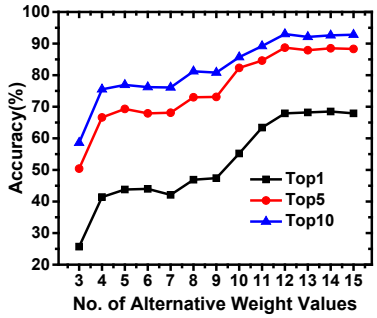

Fig.9 Classification accuracy as a function of the number of alternative weight values used in the quantization scheme.

Fig.10 presents the adverse effect of RRAM conductance variation on classification accuracy. In order to get close to the accuracy of the full-precision network, conductance variation lower than 10% is required, which is a strict restriction on RRAM characteristics. Although the variation of high conductance states is typically lower than the variation of low conductance states, the tested variations of two relatively high conductance states in our multilevel RRAM are 14.5% and 10.7%, which will lead to about 10% accuracy decrease of the VGG-16 network. In addition, the fluctuation range of small networks is about ±3% when the variation level is 50%, which is much better than the ±11.5% (since there is severe variation, **Fig.10** shows only approximate results) fluctuation range of VGG-16 network with 50% variation.

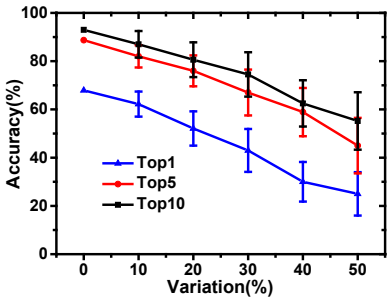

Fig.10 The influence of conductance variation on the classification accuracy of VGG-16 network.

IV. Transfer Learning

Transfer learning is a machine learning problem that focuses on using the knowledge gained from one problem to better solve a different but related problem. [28] Although transfer learning is originally proposed as a promising direction of software neural networks, it can also be used to benefit hardware RRAM-based neural network systems. Multilayer backpropagation is difficult to achieve in the RRAM-based implementation of neural networks, because the gradients will decrease after layers and become smaller than the RRAM conductance precision. Besides, complex auxiliary circuits are needed in each layer in order to calculate the update weight values. However, in transfer learning, the former layers of large-scale neural networks are fixed and only layers at the end of the network will be trained on the new database, which can naturally save the network from the 16-layer backpropagation. Moreover, for large-scale neural networks, using transfer learning can ease the restriction on RRAM characteristics, which will be analyzed in the next parts.

In our simulation, ImageNet is selected as the original dataset. And we choose the Oxford flowers dataset as the aim of transfer learning, which contains 1360 images with size about 500×500. There are 17 different categories of flowers in this dataset. In order to achieve this transfer learning task, two schemes are proposed, and explained in details.

A. Using Multilevel RRAM with Extra Memory

The transfer learning results of VGG-16 neural network is shown in **Fig.11**. In this scheme, the former 15 layers of VGG-16 network are imitated by multilevel RRAM with 1:3:8 conductance states. Additional memory is needed to store the full-precision weight values in the transfer learning layer, and the RRAM crossbar array will be refreshed by quantifying the weight array before every iteration.

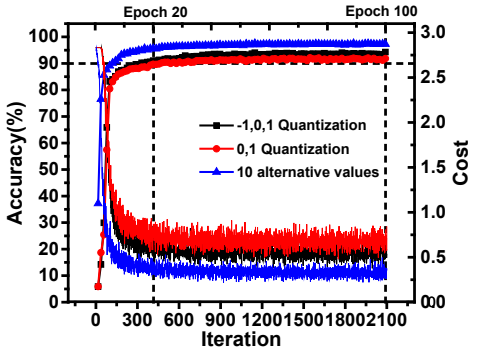

Fig.11 Using multilevel RRAM for transfer learning tasks. The blue, black and red curves correspond to different quantization schemes.

Utilizing multilevel RRAM with different number of conductance states leads to different quantization schemes. **Fig.11** illustrates the results of using two binary RRAM, one binary RRAM, and two multilevel RRAM with 1:3:8:12 conductance states, respectively, as one weight value in the transfer learning layer. First, we can see that the final classification accuracy of 10-value scheme is slightly better than that of the two-binary-RRAM scheme. However, even if we use only one binary RRAM to represent a weight value in the transfer learning layer, decent classification accuracy can still be obtained. Moreover, when the number of alternative weight values is large, the training speed becomes higher, namely less iterations are needed to get a decent classification accuracy. By the way, huge training database is often randomly divided into

small groups consists of like 64 images, which are called mini-batches. An iteration corresponds to the training using one mini-batch while an epoch corresponds to the training using the whole database and thus contains lots of iterations. Since the majority of time delay in RRAM-based hardware neural networks is the time delay of applying writing operations on numerous RRAM devices in the weight update process, using less iterations for training means a higher speed. For example, in order to get 90% accuracy, the 10-alternative-value scheme needs only 126 iterations, which is 2.7 times faster than the 0,1 quantization scheme using one binary RRAM as one weight, which takes about 475 iterations in the transfer learning process.

It should be noted that the blue, black and red curves in **Fig.11** are obtained based on the same learning rate, with the aim of controlling variables. Based on our simulations, the learning rate of this scheme can be set to a relatively large value. **Fig.12** shows results of the -1,0,1 quantization scheme with different learning rate. Although high training speed can be obtained by increasing the learning rate, the level of fluctuation will rise and the final accuracy may even decline if the learning rate becomes too high. By the way, learning rate between 2 to 5 might be appropriate in this case.

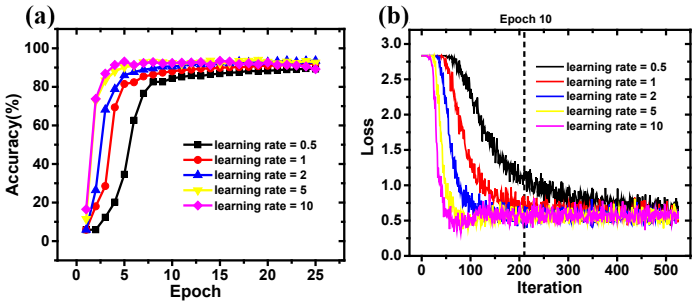

Fig.12 Training curves of the -1,0,1 quantization scheme with different learning rate. The curves in (a) show the change of accuracy during training, while the related changes of loss are shown in (b).

The operation of multilevel RRAM is simpler and much faster than that of the analog RRAM, which is one advantage of this scheme. Besides, every kind of RRAM can be used in this scheme, which enables us to optimize other factors, such as the prime cost of fabrication, the speed of RRAM and the endurance, rather than only focus on the multilevel or analog characteristics. The disadvantage of this scheme is that auxiliary circuits and extra memory are needed to assist the RRAM-based hardware neural networks.

B. Using Analog RRAM Array

In this scheme, we use analog RRAM array as the last layer with all other layers imitated by multilevel RRAM and fixed. Since the conductance states are, to some extent, continuous, the weight update process can be applied directly to the analog RRAM array without extra memory. Besides, since the typical set process of RRAM device is abrupt, we can only use the conductance states obtained by gradual reset process, which means we don't have the ability to increase the weight values in the transfer learning process. However, if we use the difference of two analog RRAM conductance to represent one weight value, then we can increase the weight by reducing the subtrahend while decrease the weight by reducing the minuend. The range of weights will double and become zero-centered by using two analog RRAM as one weight, however this method will also double the hardware cost of the transfer layer. Moreover, we can also use the set-reset pairs shown in **Fig.3 (b)**, while the time-consuming pulses will slow down the training process and the instability will decline the speed as well as the final classification accuracy.

Fig.13 presents the training process of four different mapping schemes. If the analog characteristics of RRAM device are given, a scheme is needed to map those conductance states to a range of software weight values. For example, if the analog RRAM has 200 conductance states, then using two RRAM as one weight makes 400 states available. If we map those 400 conductance states to range [-2, 2], the precision of weights would be 0.01, and the related training process is shown as the black curve in **Fig.13**. As we can see, 99.2% classification accuracy can be achieved finally.

Given a transfer learning task and specific RRAM characteristics, there is a trade-off between enlarging the range and increasing the precision of weight values. Two more schemes are developed in order to show the influence of range and precision. In the simulation of the pink curve, the range of weight values is still [-2, 2], while the precision becomes 0.1 since only 40 conductance states are utilized to represent weights. The final performance of this scheme is only 85.1%, and the speed of training is relatively low, because if the precision is not enough, many values in ΔW which are smaller than the precision can't be updated to the RRAM array in one iteration. Besides decreasing the training speed, this problem can even stop the learning process after many iterations since the error values become smaller after a period of training, which makes ΔW smaller and even more difficult to update. By increasing and optimizing the learning rate, the training process relating to the red curve can be obtained, which has the same performance as the black curve while uses only 40 conductance states.

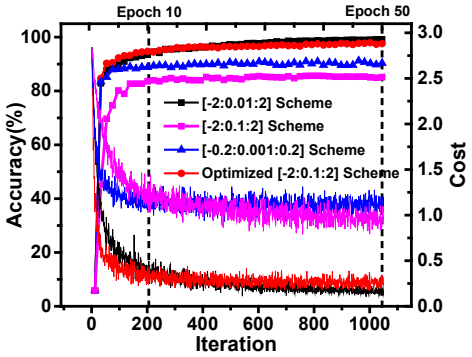

Fig.13 The training processes of four different mapping schemes using analog RRAM in the transfer learning layer.

The influence of the weight value range can be seen from the blue curve, where 400 conductance states (the same as those used in the black curve) are mapped to a range of [-0.2, 0.2]. That is to say given the same conductance states, in the blue curve, we sacrifice the range of weight values in order to get 0.001 precision. However, the final classification accuracy of this scheme is only 90.4%, which will be explained in **Fig.15**. In addition, we can see from **Fig.13** that the training speeds of the blue curve and the black curve are the same when the number of iterations is small. The reason is that the values in ΔW that are smaller than 0.01 are not very important in the training process, thus 0.01 is a sufficient precision and the training won't be accelerated by increasing the precision to 0.001.

Fig.14 shows the frequency histogram of the weights in the transfer learning layer, relating to the black curve. Based on **Fig.14**, the total amount of small reset pulses used in the transfer learning process is about 1.7×10^6. If we choose 100ns as pulse width, the time delay of this transfer learning process will be around 170ms. Moreover, the counts of 0 value is about 27k, which is much larger than the counts of other values. That is to

say the transfer layer has potential for more training iterations, since most of the values are away from their extremes.

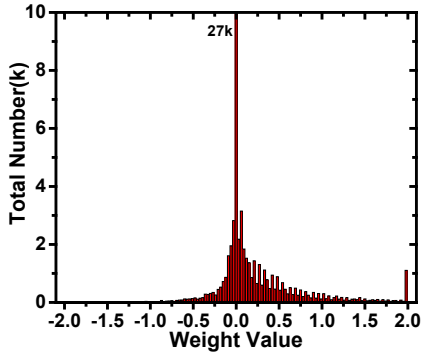

Fig.14 The frequency histogram of weights in the transfer learning layer, using the [-2:0.01:2] mapping scheme.

The frequency histogram presented in **Fig.15** corresponds to the [-0.2:0.001:0.2] quantization scheme, namely the blue curve in **Fig.13**. Since most of the weights have come to their extremes, the transfer layer becomes saturated after hundreds of iterations. That's the reason why the classification accuracy stops increasing after about 100 iterations in the blue training process. Although selecting a small mapping range can increase the precision using the same number of conductance states, it will decrease the learning capability of our transfer system and thus isn't an appropriate scheme.

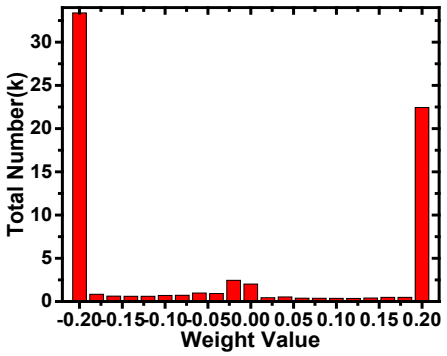

Fig.15 The frequency histogram of weights in the transfer learning layer, using the [-0.2:0.001:0.2] mapping scheme.

V. Conclusion

In this work, both small and large RRAM-based hardware neural networks are simulated based on multilevel and analog characteristics. We examine the influence of factors, such as the variation of RRAM conductance and the quantization methods, on the final performance. Transfer learning is utilized to lighten restrictions on RRAM characteristics when they are used in large-scale neural networks. In addition, two schemes to achieve transfer learning are proposed and analyzed in details.

References

[1] Lecun, Yann, Y. Bengio, and G. Hinton. "Deep learning." Nature 521.7553(2015):436-444.

[2] Ren, Shaoqing, et al. "Faster R-CNN: towards real-time object detection with region proposal networks." International Conference on Neural Information Processing Systems MIT Press, 2015:91-99.

[3] Sundermeyer, Martin, R. Schlüter, and H. Ney. "LSTM Neural Networks for Language Modeling." Interspeech 2012:601-608.

[4] Akopyan, Filipp, et al. "TrueNorth: Design and Tool Flow of a 65 mW 1 Million Neuron Programmable Neurosynaptic Chip." IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems 34.10(2015):1537-1557.

[5] Han, R. Z., et al. "A Novel Convolution Computing Paradigm Based on NOR Flash Array with High Computing Speed and Energy Efficient." 2018 IEEE International Symposium on Circuits and Systems (ISCAS).

[6] Eryilmaz, Sukru B., et al. "Brain-like associative learning using a nanoscale non-volatile phase change synaptic device array." Frontiers in Neuroscience 8.8(2014):205-205.

[7] Yu, Shimeng, et al. "Binary neural network with 16 Mb RRAM macro chip for classification and online training." Electron Devices Meeting IEEE, 2017:16.2.1-16.2.4.

[8] Ielmini, D., et al. "Neuromorphic computing with hybrid memristive/CMOS synapses for real-time learning." IEEE International Symposium on Circuits and Systems IEEE, 2016:1386-1389.

[9] Vincent, A. F., et al. "Spin-transfer torque magnetic memory as a stochastic memristive synapse for neuromorphic systems." IEEE Transactions on Biomedical Circuits & Systems 9.2(2015):166.

[10] Wong, H. S. Philip, et al. "Metal–Oxide RRAM." Proceedings of the IEEE 100.6(2012):1951-1970.

[11] Wang, Yu, et al. "Energy Efficient RRAM Spiking Neural Network for Real Time Classification." Edition on Great Lakes Symposium on Vlsi ACM, 2015:189-194.

[12] Huang, Peng, et al. "Binary Resistive-Switching-Device-Based Electronic Synapse with Spike-Rate-Dependent Plasticity for Online Learning." ACS Applied Electronic Materials 1.6 (2019): 845-853.

[13] Dong, Zhen, et al. "Convolutional Neural Networks Based on RRAM Devices for Image Recognition and Online Learning Tasks." IEEE Transactions on Electron Devices 66.1 (2018): 793-801.

[14] Duan S, et al. Small-world Hopfield neural networks with weight salience priority and memristor synapses for digit recognition[J]. Neural Computing & Applications, 2016, 27(4):837-844.

[15] Zhou, Z., et al. "The Characteristics of Binary Spike-Time-Dependent Plasticity in HfO2-Based RRAM and Applications for Pattern Recognition." Nanoscale Research Letters 12.1(2017):244.

[16] Li, Haitong, et al. "Hyperdimensional computing with 3D VRRAM in-memory kernels: Device-architecture co-design for energy-efficient, error-resilient language recognition." Electron Devices Meeting IEEE, 2017:16.1.1-16.1.4.

[17] Tang, Tianqi, et al. "Energy efficient spiking neural network design with RRAM devices." International Symposium on Integrated Circuits IEEE, 2015:268-271.

[18] Zhu, Dongbin, et al. "Resistive random access memory and its applications in storage and nonvolatile logic." Journal of Semiconductors 38.7(2017):22-34.

[19] Chen, Z., et al. "Optimized learning scheme for grayscale image recognition in a RRAM based analog neuromorphic system." IEEE International Electron Devices Meeting IEEE, 2015:17.7.1-17.7.4.

[20] Z. Dong, et al. "RRAM-based Convolutional Neural Networks for High Accuracy Pattern Recognition and Online Learning Tasks." Silicon Nanoelectronics Workshop IEEE, 2017.

[21] Russakovsky, Olga, et al. "ImageNet Large Scale Visual Recognition Challenge." International Journal of Computer Vision 115.3(2015):211-252.

[22] Krizhevsky, Alex, I. Sutskever, and G. E. Hinton. "ImageNet classification with deep convolutional neural networks." International Conference on Neural Information Processing Systems Curran Associates Inc. 2012:1097-1105.

[23] Simonyan, Karen, and A. Zisserman. "Very Deep Convolutional Networks for Large-Scale Image Recognition." Computer Science (2014).

[24] Szegedy, Christian, et al. "Going deeper with convolutions." (2014):1-9.

[25] He, Kaiming, et al. "Deep Residual Learning for Image Recognition." (2015):770-778.

[26] Song Han, Huizi Mao, and William J. Dally. "Deep Compression: Compressing Deep Neural Networks with Pruning, Trained Quantization and Huffman Coding." Fiber 56.4(2015):3--7.

[27] Rastegari, Mohammad, et al. "XNOR-Net: ImageNet Classification Using Binary Convolutional Neural Networks." (2016):525-542.

[28] Pan, Sinno Jialin, and Q. Yang. "A Survey on Transfer Learning." IEEE Transactions on Knowledge & Data Engineering 22.10(2010):1345-1359.