WEIGHT OPTIMIZATION RESEARCH FOR LOW COST HARDWARE

Thach Kim Nguyen  
Faculty of Electrical and Electronics Engineering  
HCMC University of Technology and Education HCM City, Vietnam  
nguyenkimthach.a@gmail.com

Son Ngoc Truong  
Faculty of Electrical and Electronics Engineering  
HCMC University of Technology and EducationHCM City, Vietnam  
sontn@hcmute.edu.vn

*Abstract: Currently, with the strong development of computer vision and IOT systems, the problem of saving costs, energy, and computing resources in low-cost applications is being pursued by many projects. care and attention. In this paper, the topic proposes a solution to optimize the weight set of the neural network, which is to quantize most of the weights in the form of signed binary weights 1 and -1 to apply to cheap hardware such as the Raspberry Pi Zero. Binary weights will make the calculation faster, saving energy during training compared to 32-bit weights. With the license plate data set, GTSRB was trained with the TensorFlow artificial neural network library, which achieved an accuracy of 96.01%. In addition, the study also trained the BCNN network with other data sets and achieved the accuracy of Mnist (99.81%), Cifa-10 (93.59%), Mnist-fashion (96.35), and product case code dataset CNPB (99.89%) When training on the data sets Mnist, Mnist-fashion, Cifa-10, GTSRB, and CNPB, the inner set of the BCNN neural network, the storage capacity of the BCNN binary neural network weight set decreased from 6.16 to 7.59 times compared to that of the BCNN. Conventional CNN network model with the same structure. The processing speed of the BCCN model is 5.2 times faster than a conventional CNN network with a similar structure when deployed on low-profile hardware, the Raspberry Pi Zero, with the CNPB dataset.*

# INTRODUCTION

Artificial neural networks (ANN) and deep learning with high accuracy have been gaining more and more achievements in many fields, such as image recognition, speech recognition, and prediction [1–7]. However, ANN networks are computationally expensive. The reason is that ANNs are built on a 32-bit system, and they include a very large number of computational tasks and internal parameters.

Some low-cost practical applications, such as traffic sign recognition, hand gesture recognition, product code recognition, etc., do not need to be deployed on systems with large hardware and heavy use. Using a binary neural network with quantized weights from 32-bits to 1-bit (+1 and -1) is one of the effective solutions to this problem. [8–11]. As a result, in the same network structure, the storage capacity of the BNN network is only 1/7.59 times that of the CNN network.

Compared with the DNN network, the BNN bit manipulation operations significantly improve energy efficiency [12]. The goal of the thesis in this work is to develop a low-cost hardware-based binary neural network framework for the Raspberry Pi Zero for on-box code recognition. Since the 32-bit weights have been quantized to binary, we can use the logical NOT and AND operations to replace the usual computations like work, subtraction, multiplication, and division. This will greatly reduce the training time and increase the processing speed of the neural network.

The proposed binary neural network implemented on the Raspberry Pi Zero board recognizes codes for product classification to avoid confusion.



Figure 1. Block diagram of the recognition controller using the Raspberry Pi. hardware.

Figure 1 shows a schematic representation of the control operation for the identification application. The identification program is performed on the Raspberry Pi Zero 2W board. This model will perform the task of pattern recognition, specifically the image recognition of the code on the product box.

# RELATED WORK

## Binarized Neural Networks

[18–20] are topics that have trained weights for neural networks with 32-bit floating point numbers. Intending to save resources for both the hardware and the computation process, [12] proposed a method that uses the value weights 1 and -1 to train the neural network to reduce memory consumption and hits when implementing low-cost hardware. Binary weights of 1 bit and binary activation can be substituted for 32-bit numbers to train the neural network, which will increase the network's performance by eliminating redundant elements in the network. neural convolution simultaneously helps the network achieve higher accuracy than [21]. [22] showed that CNNs with binary weights can use bit operations efficiently and speed up the network's processing speed.

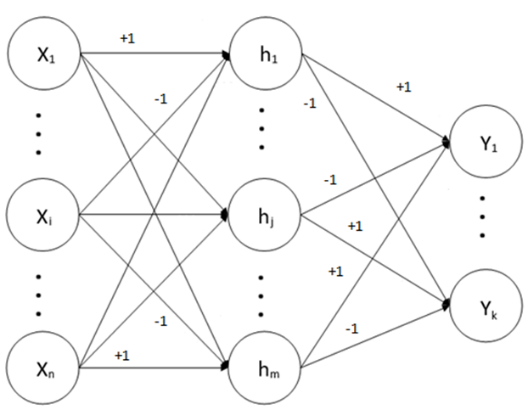


Figure 2. The diagram of the third-order neural network model, where the weights of the neural network are - 1, + 1.

Figure 2 shows a binary neural network where the neural network weights are -1 or +1. To store -1 or +1 weighted values, a memory of 8 bits can be used instead of 32 bits (typically used in full-precision neural networks) to reduce the storage space for the parameters of the model number of the network. Therefore, the capacity of a binary neural network is much smaller than that of a high-precision neural network (more than 6-7.59 times smaller). Since the capacity to store the set has been reduced, this neural network can be implemented on a low-profile embedded system such as the Raspberry Pi Zero board.

To improve accuracy, the thesis replaces the conventional CNN network with a CNN network with binary quantized weights. The topic also uses the binary activation function as the symbolic function. Therefore, the structure of the B-CNN network will be given in Figure 3 below.

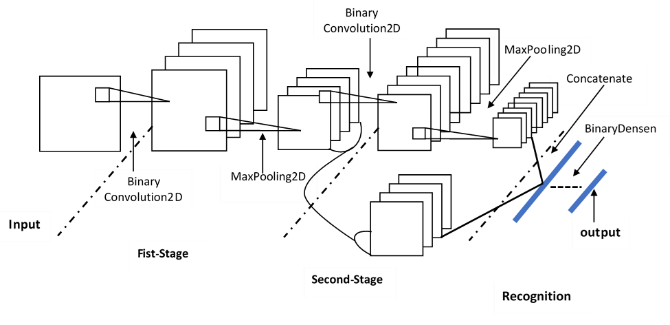


Figure 3. The architecture of the B-MNN has 7 convolutional layers (of which the 1st layer has 2 layers and the 2nd layer has 3 layers), followed by 2 fully connected layers with a set of numbers quantized into binary.

## Deterministic Binarization

When training the BNN, we will quantize the number to +1 or -1. Those two values are very convenient from a hardware perspective. To transform variables with real values into those two values, we will have to use two different functions (1).

(1)

Where is the binary quantized value (trigger or weight) and x is a real value. It works so well in practice because it's so easy to do.

## Gradient calculation and propagation algorithm

Since the derivative of the function (1) is always zero, the weights will not be suitable for the backpropagation method. To solve the above problem in this study, we use the linear propagation estimation method for gradient propagation:

q = Sign(x), (2)

Assuming that the value of has been obtained through the method of steep propagation estimation, we will calculate gr through the formula:

, (3)

Where is the slope value of q, and similarly, will be the slope value of x. If the absolute value of r is greater than 1, then the value of is now 0; otherwise, will be equal to gq.

The propagation algorithm of this topic is referred to in [23], so the topic will not be shown in detail here.

# Implement B-CNN network on multiple data sets

## GTSRB dataset

This is a dataset that includes more than 50,000 images of traffic signs. The images in this dataset range in size from 15x15 pixels to 222x193 pixels. In this dataset, the ratio between the number of images for training and the number of images for evaluation is 3:1. Namely 12630 images for the training set and 3909 images for the evaluation dataset.



Figure 4. Random samples in the set GTSRB

## MNIST dataset

The MNIST database (English: MNIST database, abbreviated from Modified National Institute of Standards and Technology database) is a large database containing handwritten digits commonly used in training image processing systems. for different photos. This dataset has about 60000 training images and 10000 evaluation images, including 10 types of numbers. This database is also widely used for training and testing in the field of machine learning.



Figure 5. The figure of random samples in the set MNIST

## MNIST Fashion dataset

Is a fashion dataset taken from articles from the fashion brand Zalando. This dataset has about 60000 training images and 10000 review images, covering 10 categories of fashion items and accessories, namely t-shirts, polo shirts, shirts, pants, skirts, jackets, sandals, sneakers, boots, and bags. Each image in the MNIST Fashion set is 28x28, with each pixel having a value between 0 and 255.

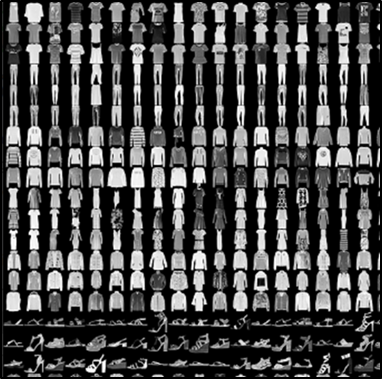


Figure 6. Random samples in the MNIST Fashion dataset

## CIFAR- 10 dataset

The dataset consists of 60000 32x32 color images divided into 10 layers, equivalent to 6000 images for each layer. The ratio of training and evaluation images is 5:1, namely, 50,000 training images and 10,000 evaluation images. This dataset is divided into five training sessions and one test, each with 10,000 images.

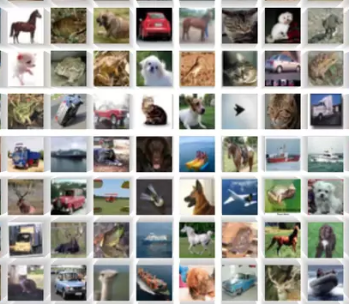


Figure 7. Image of random images in the set CIFAR-10

## CNPB dataset

The CNPB dataset (English: CNPB dataset, abbreviated from Code Number on Product Boxes) was created by us. It consists of 20,000 images with a size of 100 by 100 pixels, including 16,000 images for training. model and 4000 images for model testing. In this dataset, we have 2500 images for each code; the codes are "F06," "K11," "S19," and ' V22." These codes have sections where the number corresponds to the ordinal number of the specialized production lines, and the letter has a position in the 24-letter table corresponding to the ordinal number of the production line.





Figure 8. Patterns in the product case number data set

# EXPERIMENTAL RESULTS

## GTSRB dataset

Evaluation results of the neural network BCNN for the GTSRB traffic sign dataset, including 12630 images. The accuracy of the model is 96.01%. We have also given the graph of the change of accuracy and loss parameters during training presented in Figure 10.

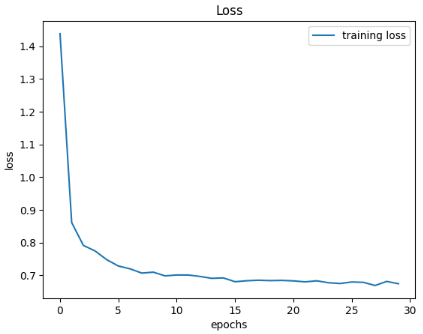


Figure 9. B-CNN recognizer training: loss vs. epochs for the GTSRB dataset

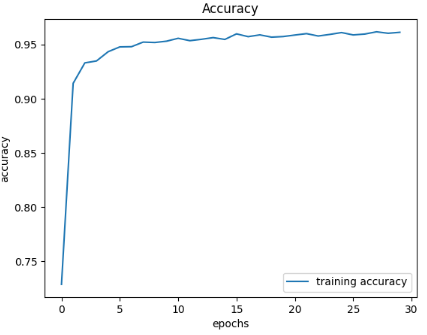


Figure 10. B-CNN recognizer training: accuracy vs. epochs for the GTSRB dataset

## MNIST dataset

Evaluation results of CNN neural network for the handwriting data set, including 70000 images. The accuracy of the model is 99.81%. We have also given the graph of the change of accuracy and loss parameters during training presented in Figure 12.

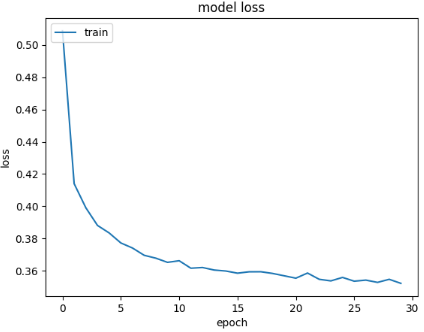


Figure 11. B-CNN recognizer training: loss vs. epochs for the MNIST dataset

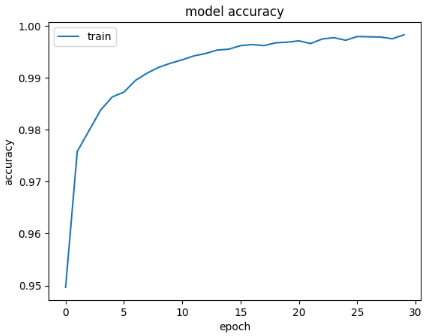


Figure 12 B-CNN recognizer training: accuracy vs. epochs for the MNIST dataset

## MNIST Fashion

Evaluation results of the neural network BCNN for the MNIST fashion items data set, including 70000 images. The accuracy of the model is 96,35%. We have also given the graph of the change of accuracy and loss parameters during training presented in Figure 14.

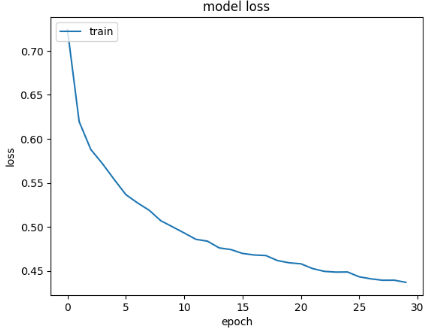


Figure 13. B-CNN recognizer training: loss vs. epochs for the MNIST Fashion dataset

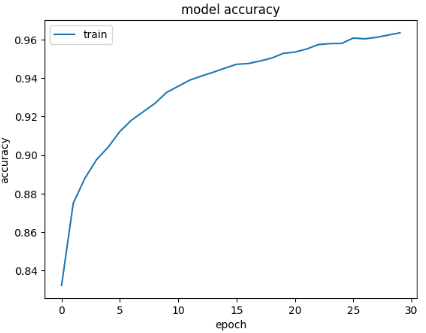


Figure 14. B-CNN recognizer training: accuracy vs. epochs for the MNIST Fashion dataset

## CIFAR- 10 dataset

Evaluation results of the BCNN neural network against the standard data set CIFAR-10, including 60000 images. The accuracy of the model is 93,59%. We have also given the graph of the change of accuracy and loss parameters during training presented in Figure 16.

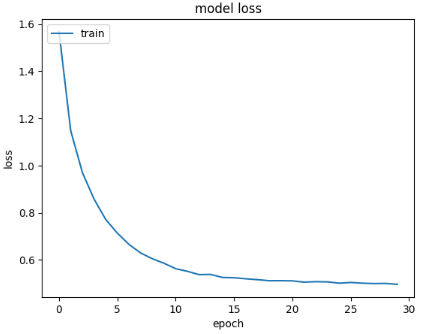


Figure 15. B-CNN recognizer training: loss vs. epochs for the CIFAR- 10 dataset

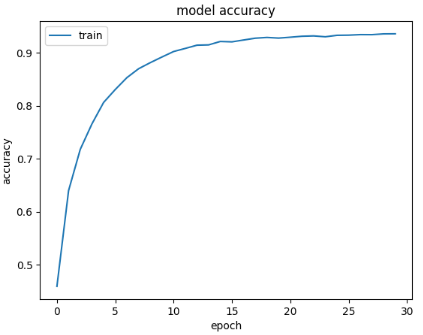


Figure 16. B-CNN recognizer training: accuracy vs. epochs for the CIFAR- 10 dataset

## CNPB dataset

The results of the BCNN neural network evaluation for the code data set on the CNPB product bin consist of 20,000 images. The accuracy of the model is 99,89%.

To test the performance, we implemented B-CNN and CNN networks on Raspberry Pi Zero 2W hardware with 500MB of RAM. The average image processing speed for applying NN per 100 images is 47,198 seconds, and a CNN network, usually a network of the same size, has a processing speed of 245.428 seconds. We see that the processing speed of the model B CNN is 5.2 times faster than that of a regular CNN. We have also given the graph of the change of accuracy and loss parameters during training presented in Figure 18.

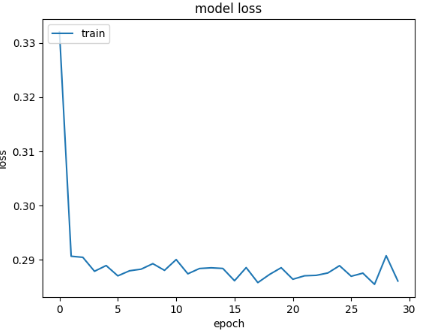


Figure 17. B-CNN recognizer training: loss vs. epochs for the CNPB dataset.

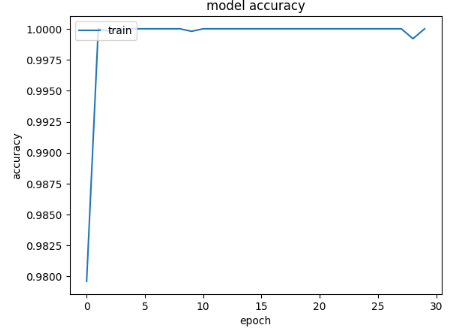


Figure 18. B-CNN recognizer training: accuracy vs. epochs for the CNPB dataset.

REFERENCES

1. Son Truong Ngoc“Low cost artificial neural network model for Raspberry Pi”, Engineering, Technology & Applied Science Research, Vol. 10, Issue 2, 2020, 5466-5469, 2020
2. A. Krizhevsky, I. Sutskever, GE Hinton, “Imagenet Classification with Deep Convolutional Neural Networks”, Advances in Neural Information Processing Systems, Lake Tahoe, USA, December 3-8, 2012.
3. BMZahran, “Using neural networks to predict the hardness of aluminum alloys”, Engineering, Technology & Applied Science Research, Vol. 5, No. 1, pp. 757-759, 2015
4. GS Fesghandis, A. Pooya, M. Kazemi, ZN Azimi, “Comparison of perceptual-based and multilayer afferent-based functional neural networks in predicting the success of new product development", Research in Engineering Science, Technology & Applications, Vol. 7, No. 1, pp. 1425-1428, 2015
5. K. He, X. Zhang, S. Ren, J. Sun, “Deep Residual Learning for Image Recognition”, in: IEEE Conference on Computer Vision and Pattern Recognition, pp. 770-778, IEEE, 2016
6. Y. Choukroun, E. Kravchik, F. Yang and P. Kisilev, "Low-bit Quantization of Neural Networks for Efficient Inference," 2019 IEEE/CVF International Conference on Computer Vision Workshop (ICCVW), pp. 3009-3018, 2019
7. R. Timofte, VA Prisacariu, LV Gool and I. Reid, “Combining traffic sign detection with 3d tracking to better assist drivers,” in Emerging topics in computer vision and other topics its application, pp. 425–446, World Science, 2012
8. SP Rajendran, L. Shine, R. Pradeep and S. Vijayaraghavan, “Real-time Traffic Sign Recognition Using yolov3-based detectors,” at the 2019 10th International Conference on Technology Computer, Communication and Networking Technology (ICCCNT), pp. 1–7, IEEE, 2019.
9. M. Coubarariaux, Y. Bengio, and J.-P. David, “Binary Connections: Training Deep Neural Networks with Binary Weights During Propagation,” in Advances in Neural Information Processing Systems, pp. 3123–3131, 2015
10. Sajad Darabi, Mouloud Belbahri, Matthieu Courbariaux, Vahid Partovi Nia, “Regular Binary Network Training”, available at: https://doi.org/10.48550/arXiv.1812.11800, 2020
11. Zhou, Shuchang & Ni, Zekun & Zhou, Xinyu & Wen, He & Wu, Yuxin & Zou, Yuheng, “DoReFa-Net: Training low bitband convolutional neural networks with low bandwidth gradients”, 2016
12. M. Courbaraux, I. Hubara, D. Soudry, R. El-Yaniv, Y. Bengio, “BinaryNet: Training Deep Neural Networks with +1 or −1 bound activations and weights”, available at: https://arxiv.org/abs/1602.02830, 2016
13. P. Sermanet and Y. LeCun, “Traffic Sign Recognition with Multi-Scale Convolutional Networks.” in IJCNN, pp. 2809–2813, 2011. Y. Yuan, Z. Xiong and Q. Wang, “Incremental Framework for Video-Based Traffic Sign Detection, Tracking, and Recognition,” IEEE Transaction on Intelligent Transportation Systems, vol. 18, no. 7, pp. 1918–1929, 2016
14. L. Deng, P. Jiao, J. Pei, Z. Wu, G. Li “GXNOR-Net: Training deep neural networks with weights and cubic activations without full precision memory in the die unified discriminant framework”, Neural Networks, Vol. 100, pages 49-58, 2018
15. H. Luo, Y. Yang, B. Tong, F. Wu and B. Fan, “Traffic Sign Recognition Using Multitasking Convolutional Neural Networks,” IEEE Transactions on Traffic Systems Smart, practice. 19, no. 4, pp. 1100-1111, 2017.
16. Y. Akhauri, "HadaNets: Flexible Quantization Strategies for Neural Networks," in IEEE/CVF Computer Vision and Pattern Recognition Workshop (CVPRW), Long Beach, CA, USA , pp. 526-534, 2019.
17. Y. Akhauri, "HadaNets: Flexible Quantization Strategies for Neural Networks," IEEE/CVF 2019 Conference on Computer Vision and Pattern Recognition Workshop (CVPRW), 2019, pp. 526-534, XNOR-Net++: Advanced Binary Neural Networks, Adrian Bulat, Georgios Tzimiropoulos, 2019.
18. R. Girshick, “Fast R-cnn,” in Proceedings of the IEEE international conference on computer vision, pp. 1440–1448, 2015.
19. W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu and AC Berg, “Ssd: Single shot multibox detector,” in European conference on computer vision, pp. 21–37, Springer, 2016.
20. S. Ren, K. He, R. Girshick and J. Sun, “Faster R-cnn: Towards Real-Time Object Detection with Area Recommendation Networks,” in Advances in Processing Systems neuroinformation processing, pp. 91–99, 2015 .
21. Y. Umuroglu, NJ Fraser, G. Gambardella, M. Blott, P. Leong, M. Jahre and K. Vissers, “Finn: A Framework for Scalable, Fast Binary Neural Network Inference,” in Proceedings of ACM/2017 SIGDA International Symposium on Field Programmable Gate Arrays, pp. 65–74, ACM, 2017.
22. C. Ma, Y. Guo, Y. Lei and W. An, “Binary Volumetric Convolutional Neural Networks for 3-D Object Recognition,” IEEE Transactions on Instrumentation and Measurement, vol. 68, no. 1, pp. 38–48, 2018.
23. X. Song, H. You, S. Zhou and W. Xie, "Traffic Sign Recognition with Binary Multi-Scale Neural Networks," the Association's 35th Youth Academic Annual Meeting 2020 China automation (YAC), Zhanjiang, China, p. 116-121, doi: 10.1109/YAC51587.2020.9337571, 2020
24. H. Qin et al., "Forward and backward information retention for precise binary neural networks," IEEE/CVF 2020 Conference on Computer Vision and Pattern Recognition (CVPR), pp. 2247-2256, 2020.
25. M. Coubarariaux, Y. Bengio, and J.-P. David, “Binary Connections: Training Deep Neural Networks with Binary Weights During Propagation,” in Advances in Neural Information Processing Systems, pp. 3123–3131, 2015.
26. J. Chen, L. Liu, Y. Liu and X. Zeng, "Learning Framework for n-Bit Quantum Neural Networks Towards FPGAs," in IEEE Transactions on Neural Networks and Systems study, practice. 32, no. 3, pp. 1067-1081, March 2021.
27. K. Hwang, W. Sung, “Designing a fixed-point relay deep neural network using +1, 0 and −1 weights", 2014 IEEE Workshop on Signal Processing Systems, Belfast, Kingdom England, October 20–22, 2014.