Binary Convolution Model for Image Classification

Edgar Solis Romeu
National Research Tomsk State University
Tomsk, Russia
solisromeu@mail.ru

Dmitry Shashev Vadimovich
National Research Tomsk State University
Tomsk, Russia
dshashev@mail.ru

***Abstract* — Binary Neural Networks present an opportunity for developing Neural Networks that require less computing power as well as energy. This is done through the use of binary values for weights and inputs. This research presents an architecture for image classification where the image and the Convolutional Layers are binarized. The result of the binary convolution is then fed to a non-binary network.**

Keywords—Binary networks, convolution, neural network, artificial intelligence, computer vision

I. INTRODUCTION

The development of neural networks has benefitted the emergence of the field of Artificial Intelligence. Convolutional Neural Networks (CNNs) have become the primary tool for resolving Computer Vision tasks, because they are highly effective for simplifying images into useful features that can then be analysed by a traditional neural network.

The adoption of computer vision is ongoing, and the number of devices that contain one type of neural network based algorithm for completing their tasks is growing. Mass produced devices like kitchen appliances, smart phones, watches, etc. can benefit from artificial intelligence features. Take for example the hypothetical case of a microwave with a camera, which is used to analyse the types of food that are being introduced into it, and then propose an ideal cooking time to the user. This is only one proposed use case to show the potential of widespread adoption of artificial intelligence enhanced devices. Even though the possibilities for improving products and their user experience are big, the producers of these devices know the limitations of implementing neural networks in appliances that traditionally don't use computers. Issues like the need for high computing power, the availability of memory and the use of energy can create constraints for the wide adoption of this type of technology. For this reason the implementation of Binary Neural Networks (BNNs) is of particular interest, since these algorithms open the possibility for developing Machine Learning systems that require less computing power and memory, as well as electrical power.

This paper will explain the proposal for a mixed CNN where the convolutional layers use a binary configuration. The resulting information is then fed into a three-layered neural network that uses non-binary numbers for training and classification. The neural network will analyse data from the MNIST data set, and recognize the number represented in the images presented as inputs. This is a fairly simple task in the field of Computer Vision, but it presents a good opportunity for exploring the benefits of Binary Neural Networks.

II. II. BINARY CONVOLUTIONAL NETWORK

A. Binary Convolutional Neural Networks

Convolutional Neural Networks have become one of the more popular algorithms for Computer Vision, more specifically for analysing images. CNNs extract simple features at a higher resolution, and transform them into more complex features with a coarser resolution[1]. These networks have multiple layers, whose input and outputs are feature maps, sets of arrays[2]. In this specific case, following the specifications of the MNIST data set, the input is going to be a two dimensional array with a width of 28 values and a height of also 28. Convolutional layers use a kernel also known as a filter. The kernel is a matrix that will move over the input data, while performing a dot product operation with the part of the input over which is passing. The result of this is a matrix of dot products. When the kernel finishes passing through the input matrix, the convolution finishes. The resulting matrix is going to be smaller than the input. The features become coarser, but the size of the overall data becomes smaller. When the convolutional layers finish, the result is an array that is used as the input for the Fully Connected Layer (FCL), which is a simple neural network with an input, hidden and output layer.

The idea of a binary neural network was first proposed by Courbariaux et al. with the name of BinaryConnect [3]. The idea was to replace multiply-accumulate operations with simple accumulations, since the first requires an important amount of space and power. Using only binary values works as a regularizer. A Binary Neural Network uses only the values 1 or -1. Usually the weights of the layers are stochastically binarized using a sign function like the one in equation 1.

$$Wb = \begin{cases} +1, \ if \ \ w \geq 0 \\ -1, otherwise \end{cases}. \qquad (1)$$

With W being the value of the weight and Wb the binarized value of it. The difference between the system proposed in this article and BinaryConnect is that the latter was not meant to be used with CNNs, and that our proposed model uses a non-binary neural network in the FC layer instead of a binary one. Nevertheless, we use the same idea when applying binarization to the image and to the kernel of the convolutional layers. The idea of the Convolutional Layer being binarized was proposed earlier by Rastergari et al.[4].

When implementing a binary network the multiplications are replaced with XNOR and Popcount operations. Fig. 1 shows the truth table of the XNOR gate. After that, a count of the number of 1s would take place and the results are given to equation 2 for the final result of the computation.

TABLE 1. XNOR TRUTH TABLE

INPUT		**OUTPUT**
A	***B***	***A XNOR B***
0	0	1
0	1	0

1	0	0
1	1	1

Fig. 1. Showing the truth table of the XNOR Gate.

$$\rho = 2 * P - N \quad (2)$$

Equation (2) calculates the result of the kernel pass over the input matrix, where P is the Popcount result and N is the total number of bits of the kernel matrix.

Overall, the architecture of the network as shown in Fig. 2 consists of two convolutions. After each convolution, the size of the image matrix gets reduced, and then goes through average pooling to simplify the content of the matrix and eliminate noise generated by the convolution. The average pooling introduces non-binary values, thus it becomes necessary to pass the matrix through a binarization process after it.

MNIST is an important data set for image classification. It contains a training set of 60000 images and a 10000 image test set. The images contained represent handwritten digits from 0 to 9. The images are in grey-scale, and contain values from 0 to 255.

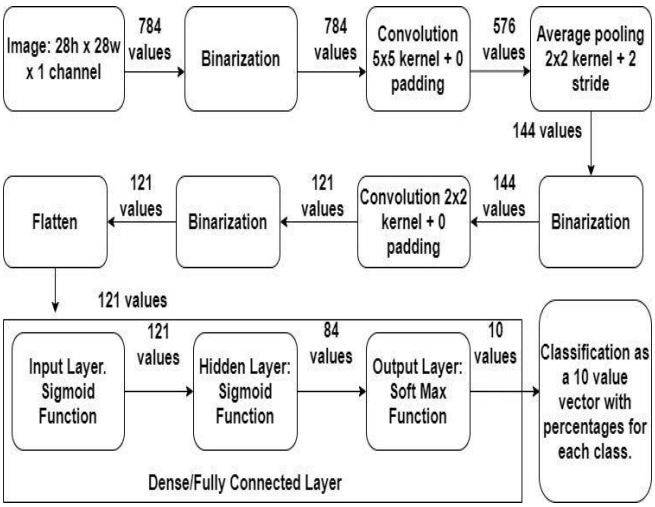

Fig. 2. Showing the architecture of the proposed Convolutional BNN. The columns to the right show how the size of the matrix gets reduced as the image goes through the layers until it becomes an array with 576 items

On Convolutional Layers the input matrix is usually given a padding to prevent over-representation and under-representation of data in the edges of the matrix, and optimise the output. The padding usually consists of extra columns and rows with zeros, but this would introduce a big distortion into a binary system, that is why padding is omitted[5]. The stride chosen for the kernel was 1. The values for the kernel consisted of binary values to probe horizontal features in the first convolution and vertical features in the second one.

The Fully Connected Layer consists of three layers: Input, Hidden and Output. The input layer has 576 neurons, the same as the final array product of the convolutional layers. The Hidden layer has 120 neurons and finally the Hidden one has 10. The initial values of the weights of the neurons are randomly selected between –0.5 and 0.5. The Output layer gives 10 outputs, which later are subjected to the Soft Max function to give the final prediction of the neuron. A sigmoid activation function is used for the Hidden layer.

B. Results

The Network was trained with 10000 samples from the MNIST. The learning rate used was of .001 and the model was trained with 200 epochs per sample. Once trained the network was tested. Using 800 data from the training data set the accuracy of the predictions was 88.5 % and testing with the validation data results 82.125%. This is an encouraging result since it shows that the network is viable.

III. Conclusion

The main objective of this research was to propose a mixed architecture that used Binary Convolutions and Non Binary Fully Connected Layers. The resulting Network achieved an accuracy of more than 82% using validation data. This shows that the resulting network can be expected to have an acceptable level of reliability, and that further improvement can be made by tweaking the values of the learning rate and the number of the epochs.

The network shows that the convolutional process can be made more efficient through binarization and that the convolved data is useful for the non-binary fully connected network.

Further research will be aimed into comparing a traditional network and this network, to determine the difference in power and memory usage. The final aim of this research is to implement a fully binary network, where both fully connected and convolutional layers are binary.

References

[1] Simard, P.Y. Best Practices for Convolutional Neural Networks Applied to Visual Document Analysis / P.Y. Simard, D. Steinkraus, J.C. Platt // Proceedings of the Seventh International Conference on Document Analysis and Recognitionю. – 2003. – Vol. 2. – P. 958-962.

[2] LeCun, Y. Convolutional Networks and Applications in Vision / Y. LeCun, K. Kavukcuoglu, C. Farabet // Proceedings of 2010 IEEE International Symposium on Circuits and Systems. – 2010. – P. 253-256.

[3] Courbariaux, M. Binaryconnect: Training Deep Neural Network with Binary Weights During Propagations / M. Courbariaux, Y. Bengio, J. David // Neural and Evolutionary Computing. – 2015. DOI: 10.48550/arXiv.1511.00363

[4] Rastegari, M. Xnor-net: Imagenet Classification Using Binary Convolutional Neural Networks / M. Rastegari, V. Ordonez, J. Redmon, A. Farhadi // Springer European Conference on Computer Vision. – 2016. – P. 525-542

[5] O'Shea, K An Introduction to Convolutional Neural Networks / K. O'Shea, R. Nash // Neural and Evolutionary Computing. – 2015. DOI: 10.48550/arXiv.1511.08458.