BINARYCONNECT : TRAINING DEEP NEURAL NETWORKS WITH BINARY WEIGHTS DURING PROPAGATIONS

Matthieu Courbariaux, Yoshua Bengio and Jean-Pierre David

Sunil Kumar Maurya (M1)
Student ID : 17M31560

Proceeding
NIPS'15 Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 2 , Pages 3123-3131

SUPPLEMENTARY PAPERS

Neural Networks with Few Multiplications

- Zhouhan Lin, Matthieu Courbariaux, Roland Memisevic and Yoshua Bengio

Binarized Neural Networks : Training Neural Networks with Weights and Activations Constrained to +1 and -1

- Matthieu Courbariaux , Itay Hubara , Daniel Soudry , Ran El-Yaniv and Yoshua Bengio

BRIEF INTRODUCTION

These papers talks about:

- Introduces binarization in neural networks and use low precision weights.
- In Forward Propagation multiplication operations are substituted by XNOR operations.
- In backpropagation, authors uses bit shift operation to do approximate calculations.

Why We don’t want Massive Multiplications ??

- ❑ Computationally Expensive
- ❑ Faster computation is likely to be crucial for further progress and for consumer applications on low-powered devices
- ❑ A multiplier free network could pave way to fast ,hardware friendly way to train neural network

BINARIZATION AS REGULARIZATION

In many cases neural network only needs very low precision

Low Precision

Stochasticity comes with benefits
-Dropout
-Noisy Gradients
- Noisy Activation functions

Stochasticity

Stochastic Binarization??

APPROACHES TAKEN

Binarize weight values

- BinaryConnect and TernaryConnect
- Binarize weights in forward/backward propagations, but store full precision version of them in the backend.

Quantize backprop

- Exponential Quantization
- Employ quantization of the representations while computing down-flowing error signals in the backward pass.

BINARY CONNECT and TERNARY CONNECT

- Weight binarization technique which removes multiplications in the forward pass.

- ❑ Consider neural network layer with N input and M output units
- ❑ Forward Propagation -> $h(Wx + b)$
- ❑ If we choose ReLU is h , then no multiplications in computing the activation function.
- ❑ Thus all multiplications reside in Wx.
- ❑ For each input vector x , N X M floating point multiplications

BINARY CONNECT

- Restricts the weights to 1 and -1
- Two ways to perform

Stochastic	Deterministic
• $P(W_{ij} = 1) = \frac{w_{ij}+1}{2}$ • $P(W_{ij} = -1) = 1 – PP(W_{ij} =1)$	• $Wij = \begin{cases} 1 \;\; if\ wij > 0 \\ -1\ if\ wij < 0 \end{cases}$

TERNARY CONNECT

- Restricts the weights to 1 , 0 and -1

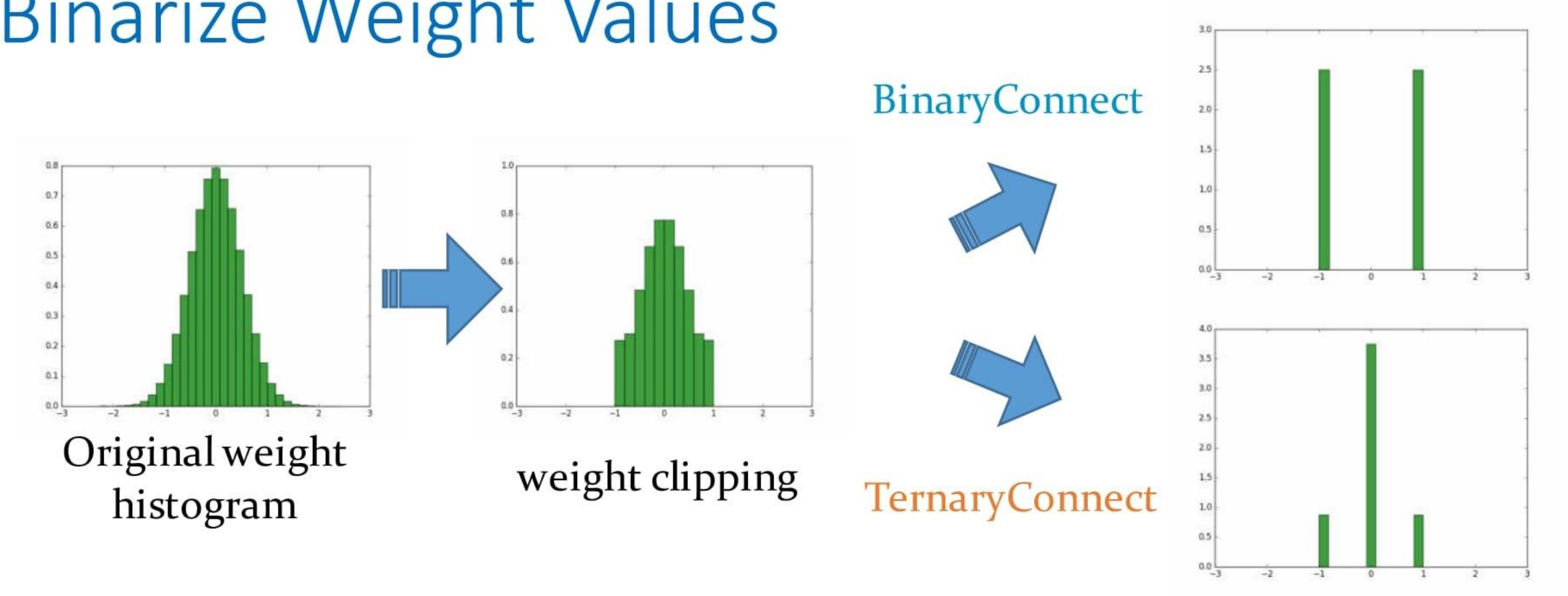
Stochastic

- If $w_{ij} > 0$:
  - $P(W_{ij} = 1) = w_{ij}$
  - $P(W_{ij} = 0) = 1 - w_{ij}$
- Else:
  - $P(W_{ij} = -1) = -w_{ij}$
  - $P(w_{ij} = 0) = 1 + w_{ij}$

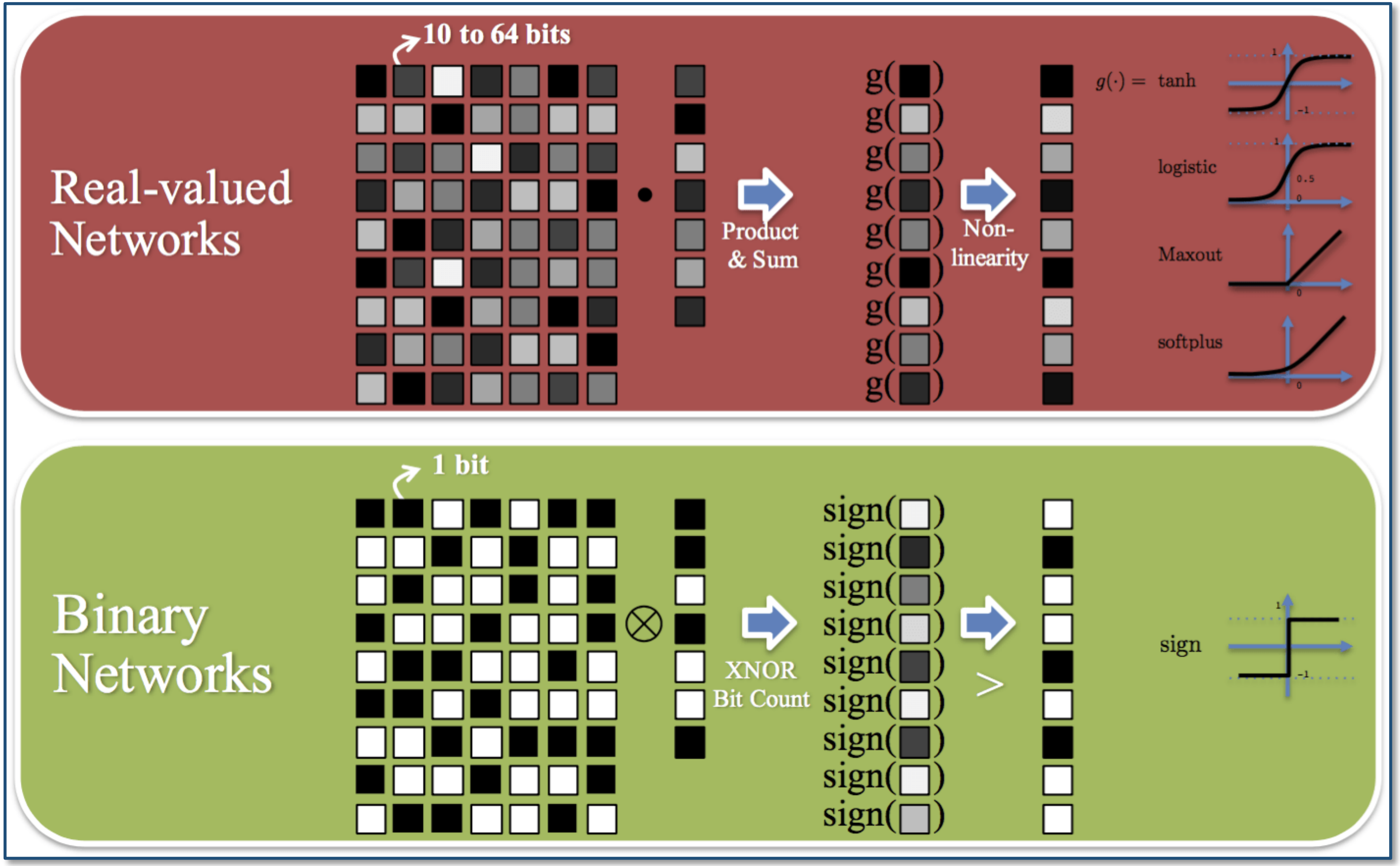
Deterministic

- Wij = $\begin{cases} 1 & w\,ij > 0.5 \\ 0 & -0.5 < wij \leq 0.5 \\ -1 & wij \leq -0.5 \end{cases}$

Binarize Weight Values



SIMPLE COMPARISON



QUANTIZED BACKPROPAGATION

- Suppose the i^{th} neural network layer with N input and M output units

  And δ = an error signal propagating from output
  Then updates for weights and biases would be

$$\Delta W = \eta \, [\delta_l \odot h'(Wx+b)] \, x^T$$

$$\Delta b = \eta \, [\delta_l \odot h'(Wx+b)]$$

η = learning rate , x = input to the layer , $\odot$ = element-wise multiplication operator

$$\delta_{l-1} = [W^T \delta] \odot h'(Wx+b)$$

ALGORITHM

Procedure :Quantized Back Propagation(model, input x, target y, learning rate η)

1. **Forward Propagation:**

 for each layer *i* in range(1,L) **do**

 $W_b \leftarrow \text{binarize}(W)$

 Compute activation a_i according to its previous layer output a_{i-1},W_b and b.

2. **Backward Propagation:**

 Initialize output layer's error signal $\delta = \frac{\partial C}{\partial a_L}$

 for each layer *i* in range(L,1) **do**

 Compute ΔW and Δb

 Update W : $W \leftarrow \text{clip}(W - \Delta W)$

 Update b : $b \leftarrow b - \Delta b$

 Compute $\frac{\partial C}{\partial a_{k-1}}$ by updating δ

EXPERIMENTS (BINARYCONNECT)

Method	MNIST	CIFAR-10	SVHN
No Regularizer	1.30 %	10.64 %	2.44 %
BinaryConnect(det.)	1.29 %	9.90 %	2.30 %
BinaryConnect(stoch.)	1.18 %	8.27 %	2.15 %
50 % Dropout	1.01 %		
Maxout Networks	0.94 %	11.68 %	2.47 %
Deep L2-SVM	0.87 %		
Network in Network		10.41 %	2.35 %
DropConnect			1.94 %
Deeply Supervised Nets		9.78 %	1.92 %

EXPERIMENTS (NN WITH FEW MULTIPLICATIONS)

Performance across different datasets:

	Full Precision	Binary Connect	Binary Connect+ Backprop	Ternary Connect + Quantized backprop
MNIST	1.33%	1.23%	1.29%	1.15%
CIFAR10	15.64%	12.04%	12.08%	12.01%
SVHN	2.85%	2.47%	2.48%	2.42%

Error rates under various implementations

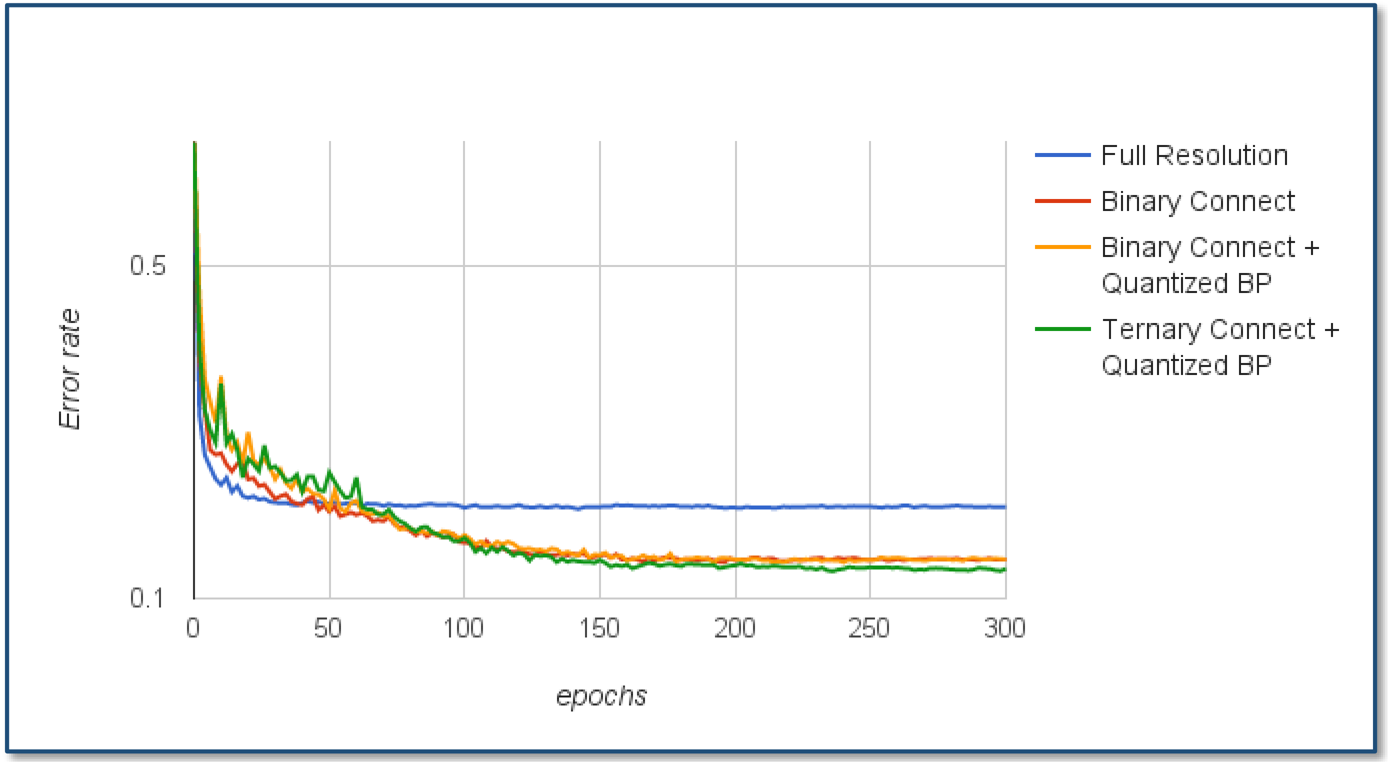
Calculation Reductions??

Estimated number of multiplications in MNIST net

	Full precision	Ternary connect + Quantized backprop	Ratio
Without BN	1.7480×10^9	1.8492×10^6	0.001058
With BN	1.7535×10^9	7.4245×10^6	0.004234

Convergence Behaviour

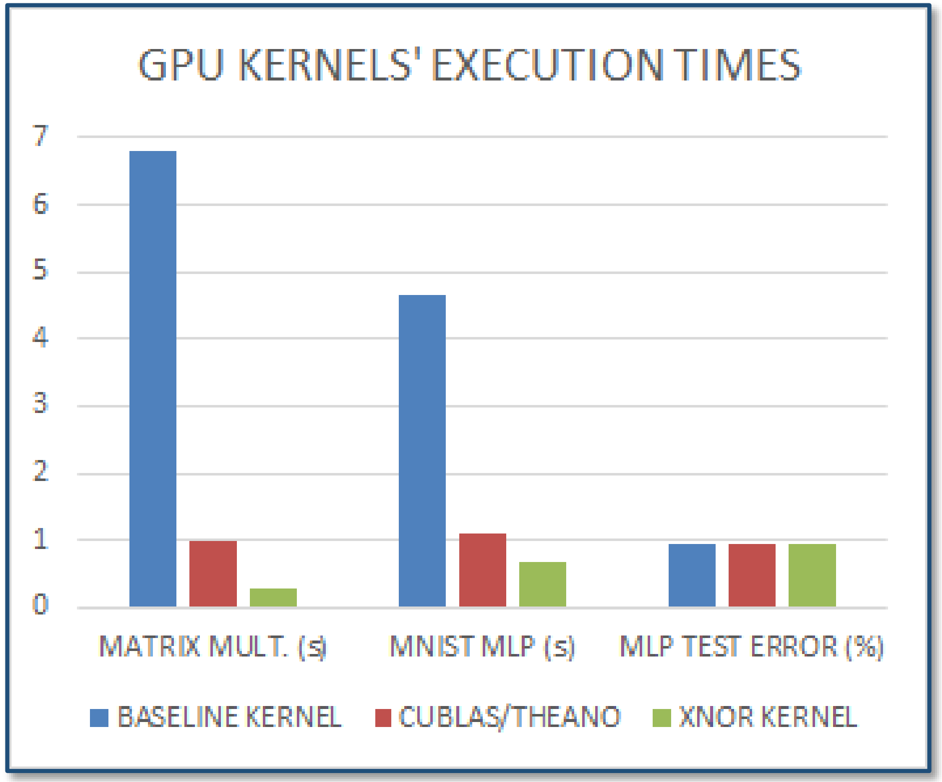
- Binarization makes the network converge slower than ordinary SGD but yields a better optimum after algorithm converges



Test error rate at each epoch, vertical axis is represented in logarithmic scale

SpeedUp with XNOR kernel

In GPUs using technique SIMD (Single Instruction Multiple Data) within a register (SWAR) , 32 binary variables can be concatenated into 32 bit registers to speedup bitwise operations (XNOR).



GPU : GTX 750 Nvidia GPU
Matrix Mult. : 8192 X 8192 X 8192 (binary) matrix multiplication

CONCLUSION

- Authors have proposed that most of the floating point multiplications can be supplanted with bitwise XNORs and left and right bit shifts during training.
- Binarization makes convergence slower but yields better optimum after convergence.
- Performance improvement attributed to regularization effect implied by stochastic sampling.
- Algorithms give good performance even with low precision weights.

REFERENCES

[1] Courbariaux, Matthieu, Bengio, Yoshua, and David, Jean-Pierre. Binaryconnect: Training deep neural networks with binary weights during propagations.arXiv preprint arXiv:1511.00363, 2015.

[2] Marchesi, Michele, Orlandi, Gianni, Piazza, Francesco, and Uncini, Aurelio. Fast neural networks without multipliers.Neural Networks, IEEE Transactions on, 4(1):53–62, 1993

[3] Machado, Emerson Lopes, Miosso, Cristiano Jacques, von Borries, Ricardo, Coutinho, Murilo, Berger, Pedro de Azevedo, Marques, Thiago, and Jacobi, Ricardo Pezzuol. Computational cost reduction in learned transform classifications.arXiv preprint arXiv:1504.06779, 2015.

[4] Burge, Peter S., van Daalen, Max R., Rising, Barry J. P., and Shawe-Taylor, John S. Stochastic bit-stream neural networks. In Maass, Wolfgang and Bishop, Christopher M. (eds.), Pulsed Neural Networks, pp. 337–352. MIT Press, Cambridge, MA, USA, 1999. ISBN 0-626-13350-4. URL http://dl.acm.org/citation.cfm?id=296533.296552

[5] Cheng, Zhiyong, Soudry, Daniel, Mao, Zexi, and Lan, Zhenzhong. Training binary multilayer neural networks for image classification using expectation backpropagation. arXiv preprint arXiv:1503.03562, 2015

[6] Nitish Srivastava ,Geoffrey Hinton,Alex Krizhevsky, Ilya Sutskever and Ruslan Salakhutdinov, Dropout : A simple way to prevent networks from overfitting*. Journal of Machine Learning Research*,15:1929-1958,2014.

THANK YOU