**Optimized Implementation of Convolution Neural Network**

**Abstract**

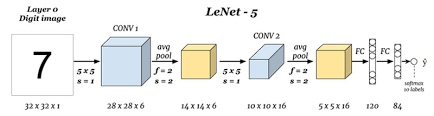
The Convolution Neural Networks have several different filters (also known as kernels) that consist of parameters that can be trained. Convolution Neural Network convolves any given image by the user spatially. It detects characteristics such as shapes, edges and corners.The large number of filters are highly efficient and effective. They learn to extract spatial features from any given image that are certainly based on the learned weights by back propagation approach. Layers of filters when stacked can be used to detect the spatial shapes which have a high level of complexity from the spatial features at every subsequent level. Therefore, the layers of filters can successfully extract the characteristics of a given image by considering the edges and vertices into an abstracted representation of high quality. Patterns in pixel values are read and extracted from the given input images in Dense Networks.

In most of the deep learning techniques, algorithms and data training is time consuming. In training neural networks most of the computation is spent on floating point multiplications. In our project, we apply an approach to training that eliminates the need for most of these. Our method consists of two parts:

First is Sparse Ternary Connect and Second one is Dual indexing module using these two parts we make a simple network which reduces computations. Experimental results across a popular datasets MNIST show that this approach not only does not hurt classiﬁcation performance but can result in even better performance than standard stochastic gradient descent training, paving the way to fast, hardware friendly training of neural networks.

**Introduction**

A Convolutional neural network (CNN) is a deep learning approach which can take an image as an input. It then assigns important learnable or trainable weights and biases to various objects in the image and is able to classify problems and other applications. Convolutional neural network is composed of multiple building blocks, such as convolutional layers, pooling layers, and fully connected layers, and is designed to automatically and adaptively learn spatial hierarchies of features through a backpropagation algorithm. The preprocessing required in a Convolutional neural network is very less as compared to other algorithms. Convolutional neural networks have the ability to learn filters(Weight and biases). A Convolutional neural network is able to successfully capture the important feature in an image through the application of relevant filters. The architecture performs a better fitting to the datasets of images due to the reduction in the number of parameters involved and reusability of weights.



In recent years, deep learning has been used extensively in a wide range of fields. In deep learning, Convolutional Neural Networks give the most accurate results in solving real world problems such as computer vision, mainly in face recognition, scene labelling, image classification, action recognition, human pose estimation and document analysis. CNN is also used in speech recognition and text classification for natural language processing.

Training deep neural networks has long been computational demanding and time consuming and requires more storage capacity and any spatial hardware. For some state-of-the art architectures, it sometimes takes weeks to get train models. Another problem is that the demand for memory can be large. For example, many models in speech recognition or machine translation need 10 Gigabytes or more of storage requirement. To deal with these issues it is common to train deep neural networks by an optimized method which we proposed or by resorting to GPU or CPU clusters and to well designed parallelization strategies .

In training a network most of the computation performed are floating point multiplications. In our project, we have tried to put more focus on eliminating most of these multiplications to reduce computation. In our project, we have combined two existing methods for optimizing CNN. These are :

1. Sparse Ternary Connect(STC) -

The proposed algorithm trains the network so that the weights are represented using a structured sparse ternary format. This format allows +1 or -1 only at specified locations, while most of the values are pruned to zero.

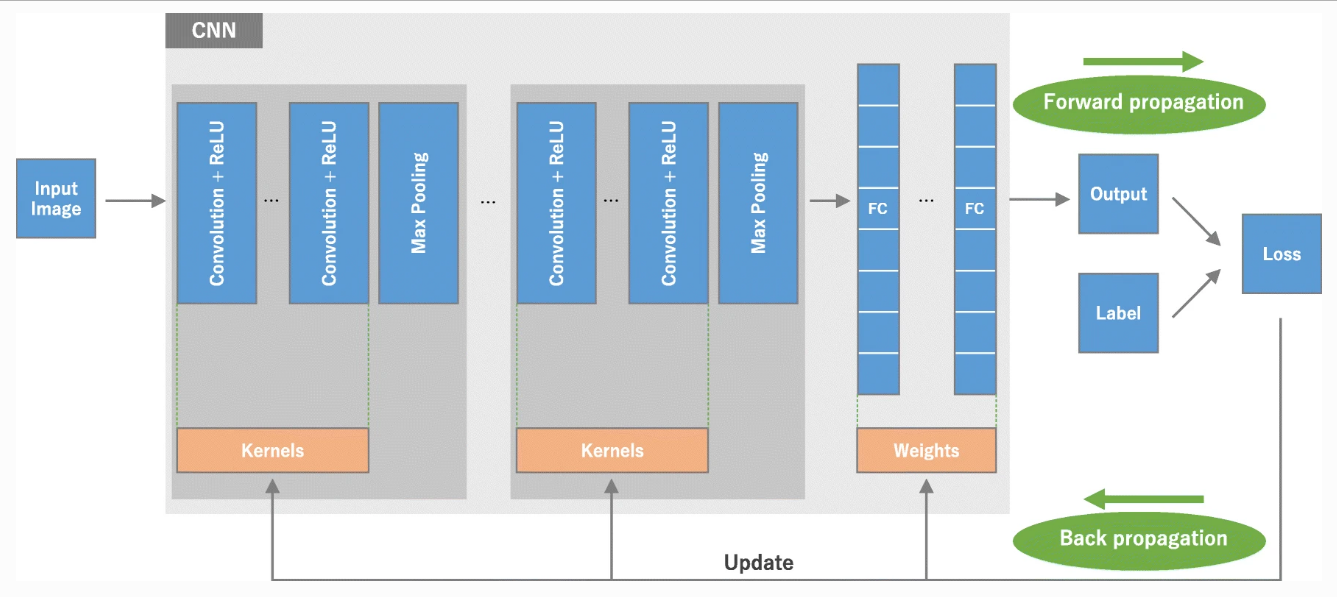
2.Dual Indexing Module(DIM) -

In convolutional neural networks, sparsity is widely observed by zeroing a large portion of both activations and weights without impairing the result, by keeping the data in a compressed-sparse format. Adding an indexing mechanism could facilitate the handling of the compressed-sparse format. Cnvlutin removes the zeros in activations and stores the memory offsets of non-zero data for rapid selection of matched weights. To reduce these calculations.

**Background and Literature Review**

**Background**

A CNN is a deep learning model used for processing data that has a grid pattern, such as images. Convolutional Neural Network is composed of multiple building blocks, they are Convolution layers, Pooling layers and Fully connected layers. The first two, i.e., convolution layer and pooling layers are used to perform feature extraction, whereas the third, i.e., a fully connected layer extracts features into final output, known as classification. A convolution layer plays a main role in CNN, which is composed of a stack of mathematical operations, such as convolution, which performs a special type of linear operation. In digital images, values of pixels are stored in the form of a two-dimensional grid, i.e., an array of numbers , and a small grid of parameters known as kernel, a feature extractor, is applied at each position of image, which makes CNNs highly efficient for image processing. As one layer provides its output to the next layer, so extracted features become more complex. The process of optimizing parameters such as kernels is called training, which is used to perform minimized difference in outputs and truth labels from an optimization algorithm known as backpropagation and gradient descent.



Above figure shows the architecture of a convolutional neural network (CNN) and the process of training. A model performance for kernels and weights is calculated from forward propagation on training dataset. Kernels and weights are updated with the help of loss value through backpropagation with the help of gradient descent algorithm using ReLU (rectified linear unit).

**Literature review**

Following articles provide the basic concepts necessary to design a CNN architecture, as well as providing the state-of-the-art energy efficient CNN designs and commenting on the future trends that still need to be addressed.

• **Neural networks with few multiplications:**

Most of the deep learning algorithms, training data is time consuming. In neural networks, most of computation is spent on floating point multiplication. Our paper consists of two methods. Firstly, we randomly binarize weights to convert multiplications involved in calculating the sign changes in hidden states. Secondly, while error in back-propagates, for binarizing the weights we calculate each layer to convert multiplication in binary shifts. Experimental results on 3 most popular datasets i.e. MNIST, CIFAR10, SVHN show that this approach does not affect classification but results in better performance than gradient descent.

• **A Novel Zero Weight/Activation-Aware Hardware Architecture of Convolutional Neural Network:**

It is essential to accelerate convolutional neural networks (CNNs) due to their ever-increasing areas from server and mobile to IoT devices. Depending on the fact CNNs are characterized by a large amount of zero values in both activation and kernel weights. So, we suggest a novel hardware accelerator for exploiting zero weights and kernels for CNNs.

• **Design Space Exploration of FPGA-Based Deep Convolutional Neural Networks:**

Deep Convolutional Neural Network (CNN) has been widely used for pattern recognition and classification. Due to the fast growing Internet of Things (IoTs) and wearable devices, it is used to implement DCNNs in portable systems. Still, novel computing is required to develop DCNNs, which have large consumption of power and have complex topologies in systems with limited areas and power supplies. To bring success to DCNNs, Stochastic Computing (SC) can completely simplify the hardware implementation of arithmetic units.

• **Supporting Compressed-Sparse Activations and Weights on SIMD-like Accelerator for Sparse Convolutional Neural Networks:**

In convolutional neural networks, sparsity is widely used in zeroing a large portion of activations and weights without changing the results. By keeping data in compressed sparse format, the consumption of energy must be less due to memory traffic. The wide SIMD-like MAC engine adopted in CNN accelerators does not support the input in compressed form due to the data misalignment.

• **Sparse Ternary Connect: Convolutional Neural Networks Using Ternarized Weights with enhanced Sparsity:**

To achieve state-of-the-art results, Convolutional Neural Networks (CNNs) are essentially used in a wide range of tasks. In this paper, we use ternary weights in both interference and training of CNNs. And then we use Sparse Ternary Connect to convert floating kermel weights values to +1, -1 and 0.

**Importance of the Project**

Convolution neural networks (CNN) are known for its high accuracy for image recognition. In recent years, it has been widely used in many image-related machine learning algorithms. CNN consists of a large number of computations and it is essential to accelerate the CNN computation by a hardware accelerator, which can be FPGA, GPU and ASIC designs. However, CNN accelerator has a drawback: the large time and power consumption caused by the data access of off-chip memory.

Optimization of CNN will help in various fields of technologies. It can be a great support to the Cyber Security and Automobile industry.

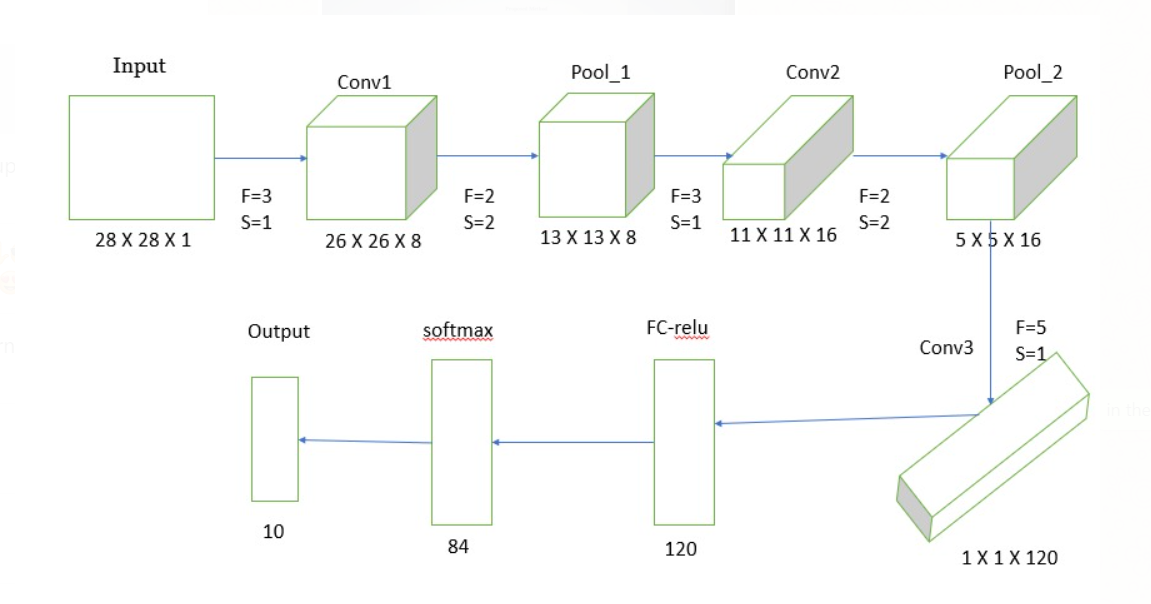
CNN has the following advantages:

* Weight sharing
* Memory Saving
* Independent of local variations in Image
* Equivariance

**Proposed Method**

We have experimented using three different architectures CNN with single layer, CNN with two layers and Lenet architecture.We have used the MNIST dataset .

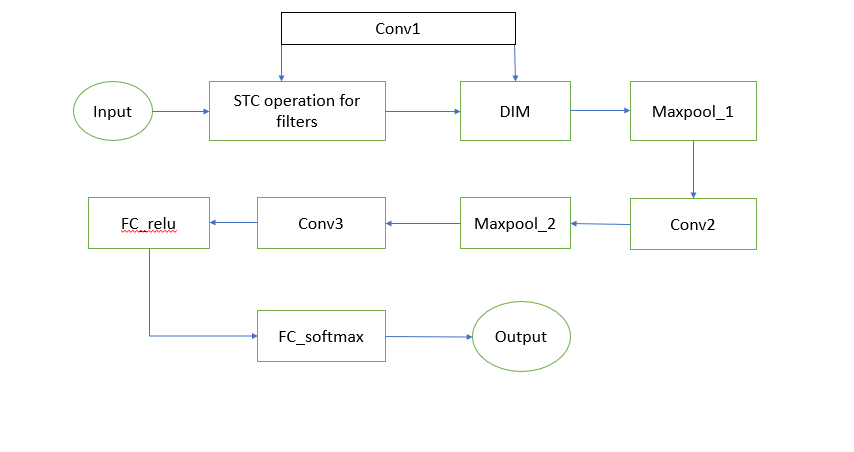
The LeNet architecture consists of the following layers:



In our proposed method first we will sparse ternerize the weight matrix .Based on the value of rho in the STC method , we will make some fixed number of filter values as zero .If the remaining kernel values which are >0 will be made +1 and those values which are <0 will be made -1.

Next , we will find the indices of the input matrix which are not zero .

**Flowchart of our project implementation:**

****

**Note: Conv2 and Conv3 are the same as Conv1 and for Convolutional we are using relu activation.**

The steps of the proposed method are-

A. Sparse ternary connect-

In Sparse ternary connect,we ternerize the weights to compress the network . IN proposed method we train the network, so that the weights are in sparse ternary format.In this format we prune most of the values to zeros and remaining values are set to be +1,-1.

In the original TC, weights can be -1, 0 and 1 ,if the weights are zero ,then we need not to consider those values for multiplication. But in the worst case it may also be possible that none of the weights are zero in Ternary Connect. So we use STC,which sets a particular number of weights to zero and ternerizes the remaining non zero weights . We use a parameter ρ (0 ≤ ρ ≤ 1), and N = |W|∗ρ weights will be set to zero where |W| is the size of the filter(or kernel). Figure A.b shows the difference between TC and STC conversions. In the original Ternary Connect,in the worst case, we need 8 adders to accumulate the 9 element-wise multiplication. for the STC with ρ =0 .5 and filter size 3\*3 ,we will make 4 elements as zeros,so for the remaining 5 non zero elements only 4 adders are needed for the accumulation.

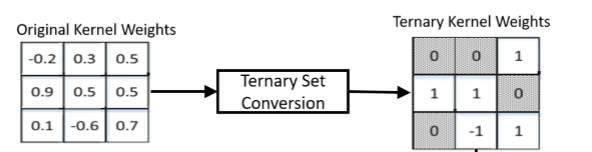


Fig A.b

Now, we will apply DIM for the kernel matrix obtained through STC.

B. Dual Indexing Module(DIM)-

In Dual Indexing Module we find out the indices of input which are not zero , then we finally take the indices whose values of both inputs and filters are not zeros . Then we will perform multiplication only on those indices in the forward propagation .

In summary, in the proposed model, we will first convert the kernel weights in sparse ternary connect and then apply the dual indexing method on ternary weights and activation values to identify the effectual weight-activation indexes.

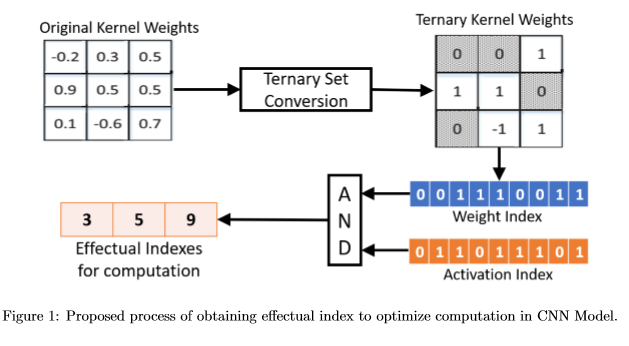


Fig B.a

**Results and discussion**

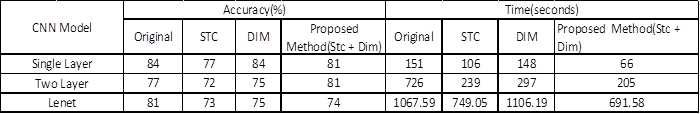
****

Fig : Comparison of Accuracy and Time for Different Approaches

In our CNN model we have used different architectures like single layer architecture, two layer architecture and Lenet-5 architecture. We have applied different optimization techniques like Sparse Ternary Connect(STC) ,Dual Indexing Module and combination of both i.e (STC +DIM) approaches. Above Fig shows the comparison between time taken and accuracy obtained for original, STC, DIM and our proposed method on MNIST dataset. Use of STC in CNN reduces the number of calculations which results in time reduction. Accuracy may decrease in some amount when we use STC because we use ternarized weights in STC. We can increase accuracy of our CNN model by using DIM technique. But from the above table we can easily see that, when we use STC and DIM together it will take less time compared to other methods without much loss in accuracy. If we consider single layer and two layer architecture, our proposed method takes less than half the time taken by original CNN. We have got these results just by applying optimization techniques only for forward propagation of CNN. In future, for better results we can further extend this method to backward propagation also.

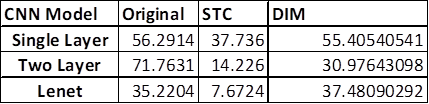
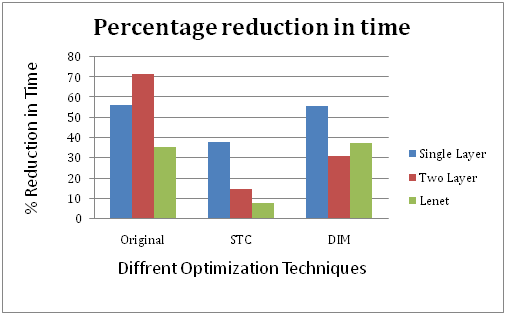


Fig : Percentage reduction in Time



From the above graph we can easily observe that for original CNN and STC we have reduced much time for Lenet-5 as compared to single layer and two layer architectures. Above results show that Lenet-5 is the most efficient CNN architecture when we use it with the STC method.

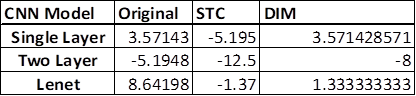


Fig : Percentage Reduction in Accuracy

From the above graph we can observe that STC method gives poor accuracy for every architecture as compared to the original CNN. The DIM method is giving the same accuracy as the original one. Two layer architecture gives worst accuracy for each approach but Lenet-5 is giving good accuracy for each approach.

**Conclusion and Future Scope**

From above observations, we can conclude that use of Lenet-5 architecture for our CNN model not only reduces computational time but also improves accuracy on the dataset. When we use our proposed method with Lenet-5 it will enhance performance of our model.

* In those results we have observed that for Original CNN, the single layer gives the output in just 151 secs. and with 84% accuracy. The two layers give the output in 726 secs. with 77% accuracy. LeNet gives the output in 1067 secs. with 81% accuracy.
* We have observed that for STC CNN, the single layer gives the output in just 106 secs. and with 77% accuracy. The two layers give the output in 239 secs. with 72% accuracy. LeNet gives the output in 749 secs. with 73% accuracy.
* We have observed that for Original CNN, the single layer gives the output in just 148 secs. and with 84% accuracy. The two layers give the output in 297 secs. with 75% accuracy. LeNet gives the output in 1106 secs. with 75% accuracy.
* The STC method gives poor accuracy for every architecture as compared to original CNN. The DIM method is giving the same accuracy as the original one. Two layer architecture giving the worst accuracy for each approach Lenet-5 is giving good accuracy for each approach.

In recent years, CNN and the work related to CNN has been rapidly increasing. It is evident that it is widely used for many computer science fields and the IT world, like gender recognition, image recognition, object recognition etc. CNN has also been widely used in various other domains across the world. It has accomplished astonishing achievements that include medical research and in radiology. In 2015, there was a research of image recognition field, where the image recognition accuracy of ResNet surpassed human accuracy i.e. ResNet accuracy was highly efficient and of higher quality than human resources. This resulted in an outstanding performance of CNN and from this time, it became more and more popular in various AI fields and applications.

Deep learning is considered as one of the most important methods to solve complex tasks. Although we can use deep learning to solve various complicated tasks and applications, it has few limitations. It is very important to have a good knowledge in the key concepts.It is important to be aware of the advantages of CNN as well as the limitations of deep learning. This will make the full use of it in different fields.

Optimizing CNN is going to help in various fields such as

* Computer Vision
* Scene Labelling
* Action Recognition
* Human Pose Estimation
* Document Analysis
* Natural Language Processing

**References**

[1] Z. Lin, M. Courbariaux, R. Memisevic, and Y. Bengio, “Neural networks with few multiplications,” arXivpreprint arXiv:1510.03009, 2015.

[2] D. Kim, J. Ahn, and S. Yoo, “A novel zero weight/activation-aware hardware architecture of convolutional neural network,” in Design, Automation & Test in Europe Conference & Exhibition (DATE), 2017. IEEE, 2017, pp. 1462–1467.

[3] M. Motamedi, P. Gysel, V. Akella, and S. Ghiasi, “Design space exploration of fpga-based deep convolutional neural networks,” in 2016 21st Asia and South Pacific Design Automation Conference (ASP-DAC). IEEE, 2016, pp. 575–580.

[4] C.-Y. Lin and B.-C. Lai, “Supporting compressed-sparse activations and weights on a simd-like accelerator for sparse convolutional neural networks,” in 2018 23rd Asia and South Pacific Design Automation Conference (ASP-DAC). IEEE, 2018, pp. 105–110.

[5] C. Jin, H. Sun, and S. Kimura, “Sparse ternary connect: Convolutional neural networks using ternarized weights with enhanced sparsity,” in 2018 23rd Asia and South Pacific Design Automation Conference (ASP-DAC). IEEE, 2018, pp. 190–195.

[6]R. Al-Rfou, G. Alain, A. Almahairi, C. Angermueller, D. Bahdanau, N.

Ballas, et al., “Theano: A Python framework for fast computation of

mathematical expressions”, arXiv preprint arXiv:1605.02688, 2016.

[7] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, et al.,

“Tensorflow: Large-scale machine learning on heterogeneous distributed

systems”, arXiv preprint arXiv:1603.04467, 2016.

[8] R. Collobert, K. Kavukcuoglu and C. Farabet, “Torch7: A matlab-like

environment for machine learning”, in BigLearn, NIPS Workshop, No.

EPFL-CONF-192376, Dec. 2011.

[9] K. Simonyan and A. Zisserman, “Very deep convolutional networks for

large-scale image recognition”, arXiv preprint arXiv:1409.1556, 2014.

[10] A. Krizhevsky, I. Sutskever and G. E. Hinton, “Imagenet classification

with deep convolutional neural networks”, in Advances in neural information processing systems, pp.1097-1105, Dec. 2012.

[11] G. Hinton, L. Deng, D. Yu, G. Dahl, A. rahman Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. Sainath, and B. Kingsbury, “Deep

neural networks for acoustic modeling in speech recognition,” Signal

Processing Magazine, 2012.

[12] L. A. Gatys, A. S. Ecker, and M. Bethge, “Image style transfer using

convolutional neural networks,” in Proceedings of the IEEE Conference

on Computer Vision and Pattern Recognition (CVPR), Jun 2016.

[13] S. Han, X. Liu, H. Mao, J. Pu, A. Pedram, M. A. Horowitz, and W. J. Dally, “EIE: Efficient inference engine on compressed deep neural net-

work,” in International Conference on Computer Architecture (ISCA), 2016.

[14] J. Albericio, P. Judd, T. H. Hetherington, T. M. Aamodt, N. D. E. Jerger, and A. Moshovos, “Cnvlutin: Ineffectual-neuron-free deep neural network computing,” in 43rd ACM/IEEE Annual International Symposium on Computer Architecture (ISCA), 2016, pp. 1–13.