Annals of R.S.C.B., ISSN:1583-6258, Vol. 25, Issue 6, 2021, Pages. 2075-2083
Received 25 April 2021; Accepted 08 May 2021.

Hash Trick Based Deep Neural Network Implementation Using Optimized Stochastic Multiplier

Dr. M.USHARANI*, R.Hemalatha, A.Prithvi**, K.B.Samruthi**, P.Logeshwari****

Assistant Professor, Dept of ECE, Velammal Engineering College, Chennai, Tamilnadu, India.

B.E, ECE, Velammal Engineering College, Chennai, Tamilnadu, India.

ABSTRACT

Image and speech recognition has been possible now with a promising technology deep neural networks (NN). Hardware implementation of the neural networks enables significant power consumption. It is principally because of non-uniform pipeline structures and inalienable excess of various number-crunching tasks that must be performed to deliver each single yield vector. This paper gives a technique to the plan of very much improved force proficient NNs with a uniform design appropriate for equipment execution with hash tricking. An error resilience analysis was acted to decide key limitations for the design of approximate multipliers that are utilized in the subsequent design of NN. By methods of a search based approximation method, approximate multipliers showing wanted tradeoffs between the exactness and execution cost were made. Huge improvement in area effectiveness was acquired in the two cases concerning regular NNs.

INTRODUCTION

Conventional camera based frameworks with PC vision handling are utilized in different business sectors e.g. Robotics,Automotive,Wearable computing, Industrial, and Mobile. Before, these frameworks utilized hand-made features(e.g. SIFT) [2] followed by a pre-trained classifier (e.g. AdaBoost) [2] to recognise objects of interests in the images. Current frameworks execute Deep learning procedures like Convolution Neural Network (CNN) for picture grouping, on account of fast advances in calculation power alongside wide-spread utilization of the ease camera giving enormous information.

Deep neural network (DNN) has made extraordinary progress in large information zone since its resurgence in 2018 [2]. Effective equipment designs for DNNs is one of the intriguing issues in both scholarly and modern social orders [1] [5] [9]-[11].The softmax layer is generally utilized in various DNNs [3]-[4] [6]. Be that as it may, most past works center around network increases enhancement in DNNs and proficient equipment execution for softmax work has not been highly examined. Productive equipment designs of softmax layer are wanted for high velocity profound learning frameworks. The exponentiation and division computations in softmax work are very costly, particularly in implanted frameworks. An equipment engineering for softmax has been proposed in [7]. Nonetheless, no definite conversation about execution of exponential and the logarithmic units, which have extremely high complexity.

Annals of R.S.C.B., ISSN:1583-6258, Vol. 25, Issue 6, 2021, Pages. 2075-2083
Received 25 April 2021; Accepted 08 May 2021.

RELATED WORK

Bai et al 2019 plan and create a hybrid structured DNN (hybrid DNN), consolidating both profundity in-space (spatial) and depth in time deep learning attributes. Our hybrid DNN utilizes memristive neurotransmitters working in a various leveled data preparing design and postponement based spiking neural network (SNN) modules as the readout layer.

Dey et al 2020 presents Deep learning (DL)- based system to roughly foresee the underlying plan of the force framework organization, thinking about various unwavering quality imperatives. The proposed structure decreases numerous iterative plan steps and rates up the absolute plan cycle.

Edstrom et al 2019 proposed a memory equipment advancement to meet the tight force spending plan in IoT edge gadgets by thinking about the security, exactness, and force productivity tradeoff in differentially proficient deep learning frameworks

Han, D et al 2019 proposed an energy proficient deep neural network (DNN) learning processor utilizing direct feedback alignment (DFA). The proposed processor accomplishes 2.2 × quicker learning speed contrasted and therefore past learning processors by the pipelined DFA (PDFA).

Mody et al 2017 propose novel systolic and completely pipelined design for convolution layer which can scale to an elite at a low region. The engineering depends on creative methods to be specific vector outer product and intelligent output feeder to enable 3 degrees of parallelism in particular information esteems, yields and contributions alongside pipelining of register components with information developments. The proposed design is versatile to give handling throughput of 64/256/512/1024 Multiplies and Add (MAC) per cycle. The engineering can approach check 600 MHz in low force 28 nm CMOS measure hub empowering execution of 1.2 Tera-Ops (TOPS).

MSebastian et al 2019 present an outline of the use of in-memory computing in deep learning, a part of AI that has altogether added to the new development in AI. The philosophy for both deduction and preparing of profound neural organizations is introduced alongside test results utilizing phase-change memory (PCM) gadgets.

Wang et al 2018 proposed a multiple algorithmic strength reduction strategies and fast addition methods to optimize the architecture. By using these techniques, complicated logic units like multipliers are eliminated and therefore the memory consumption is essentially reduced while the accuracy loss is negligible. The proposed design is coded using hardware description language (HDL) and combined under the TSMC 28-nm CMOS technology.

Zhu et al 2020 proposed a softmax work is right off the bat streamlined by investigating algorithmic strength decreases. A short time later, an equipment amicable and exactness customizable figuring strategy for softmax is proposed, which can meet diverse accuracy necessities in different deep learning (DL) errands. In light of the above advancements, a productive engineering for softmax is introduced. The proposed configuration is coded utilizing equipment depiction language and assessed on two stages, Xilinx Zynq-7000 ZC706 improvement board and TSMC 28-nm CMOS innovation, individually.

Annals of R.S.C.B., ISSN:1583-6258, Vol. 25, Issue 6, 2021, Pages. 2075-2083
Received 25 April 2021; Accepted 08 May 2021.

PROPOSED HASHNET

In this segment we present HashedNets, a novel variety of neural organizations with radically diminished model sizes (and memory requests). We initially present our methodology as a technique for irregular weight sharing across the organization associations and afterward portray how to encourage it with the hashing stunt to keep away from any extra memory overhead.

Random weight sharing

In a standard completely associated neural organization, there are (n'+1)_n'+1 weighted associations between a couple of layers, each with a relating free boundary in the weight grid V'. We expect a limited memory spending plan per layer, K' _ (n' + 1) _ n'+1, that can't be surpassed. The undeniable arrangement is to fit the neural organization inside spending plan by decreasing the quantity of hubs n'; n'+1 in layers '; '+1 or by diminishing the piece accuracy of the weight lattices (Courbariaux et al., 2014). Be that as it may if K' is adequately little, the two methodologies essentially decrease the capacity of the neural organization to sum up (see Section 6). All things considered, we propose another option: we keep the size of V' immaculate however decrease its viable memory impression through weight sharing. We just permit precisely K' various loads to happen inside V', which we store in a weight vector w' 2 RK' . The loads inside w' are shared across numerous haphazardly picked associations inside V'. We allude to the resultant matrix V' as virtual, as its size could be expanded (for example hubs are added to covered up layer) without expanding the real number of parameters of the neural network.

Figure 1 represents a neural network has one hidden layer, four input units and two output units. Associations are arbitrarily assembled into three classifications for every layer and their loads are appeared in the virtual weight matrices V1 and V2. Associations having a place with a similar shading share a similar weight esteem, which are put away in w1 and w2, respectively.Overall, the whole organization is compacted by a factor 1=4, for example the 24 loads put away in the virtual matrix V1 and V2 are decreased to just six genuine qualities in w1 and w2. On information with four information measurements and two yield measurements, a customary neural organization with six loads would be confined to a single covered up unit.

Hashed Neural Nets (Hashed Nets)

An execution of random weight sharing can be inconsequentially accomplished by keeping an secondary matrix comprising of every association's gathering assignment. Tragically, this unequivocal portrayal puts an unwanted breaking point on potential memory saving.

Annals of R.S.C.B., ISSN:1583-6258, Vol. 25, Issue 6, 2021, Pages. 2075-2083
Received 25 April 2021; Accepted 08 May 2021.

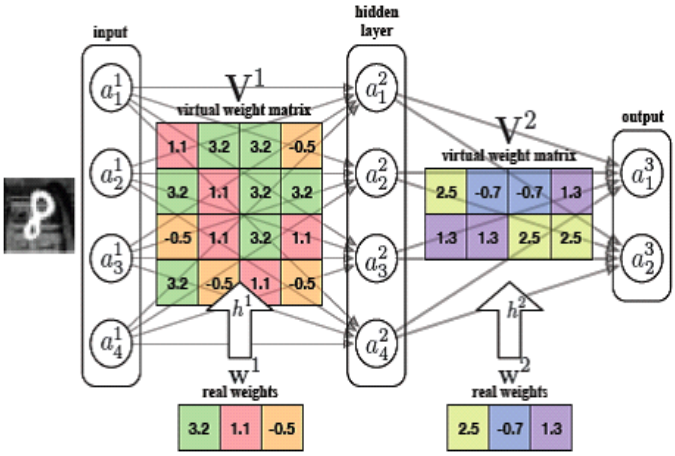

Figure 1 A representation of a neural network with arbitrary weight sharing under compression factor 1/4 . The 16+9 = 24 virtual weights are compacted into 6 real weights. The matrix elements that share the same weight are in same colours.

We propose to execute the random weight sharing tasks utilizing the hashing trick. Thusly, the common weight of every association is dictated by a hash function that requires no capacity cost with the model. In particular, we allocate to V a component of wA recorded by a hash function hA(i, j), as follows:

$$V_{ij}^{\ell} = w_{h^{\ell}(i,j)}^{\ell},$$

where the (roughly uniform) hash function hA(,) maps a key (I, j) to a characteristic number inside 1, . . . , KA . In the case of Figure1, h1(2, 1) = 1 and consequently V 1 2;1=w1=3:2. For our analyses we utilize the opensource implementation xxHash.

Feature hashing versus weight sharing

This part centers around a solitary layer all through and to improve on documentation we will drop the super-contents '. We will signify the information actuation as a = a' 2 Rm of dimensionality m=n'. We signify the yield as z=z'+1 2Rn with dimensionality n=n'+1.

To facilitate weight sharing within a feed forward neural network, substitution of equation1 and equation2

$$z_i = \sum_{j=1}^{m} V_{ij} a_j = \sum_{j=1}^{m} w_{h(i,j)} a_j .$$

On the other hand and more in accordance with past work (Weinberger et al., 2009), we may interpret HashedNets as far as feature hashing. To compute zi, we first hash the initiations

Annals of R.S.C.B., ISSN:1583-6258, Vol. 25, Issue 6, 2021, Pages. 2075-2083
Received 25 April 2021; Accepted 08 May 2021.

from the past layer, a, with the hash mapping function

φi(.)= :Rm ! RK. We at that point register the inner product between the hashed portrayal φi(a) and the parameter vector w,

$$z_i = \mathbf{w}^\top \phi_i(\mathbf{a}).$$

Both w and _i(a) are K-dimensional, where K is the quantity of hash buckets in this layer. The hash mapping function _i is characterized as follows. The kth component of φi(a), for example [φi(a)]k, is the amount of factors hashed into bucket k:

$$\begin{aligned} z_i &= \sum_{k=1}^{K} w_k \left[\phi_i(\mathbf{a})\right]_k = \sum_{k=1}^{K} w_k \sum_{j:h(i,j)=k} a_j \\ &= \sum_{j=1}^{m}\sum_{k=1}^{K} w_k a_j \delta_{[h(i,j)=k]} \\ &= \sum_{j=1}^{m} w_{h(i,j)} a_j. \end{aligned}$$

Sign factor. With this comparability between random weight sharing and feature hashing on input activations, HashedNets acquire a few valuable properties of the feature hashing. Weinberger et al. (2009) present an extra sign factor φ(i; j) to eliminate the inclination of hashed internal products because of impacts. For similar reasons we increase (3) by the sign factor φ(i; j) for defining V.

$$V_{ij} = w_{h(i,j)} \xi(i,j),$$

where E(i; j) : N - > +-1 is a subsequent hash function autonomous of h. Fusing E(i; j) to highlight hashing and weight sharing doesn't change the proportionality between them as the confirmation in the past segment actually holds with the sign term.

Annals of R.S.C.B., ISSN:1583-6258, Vol. 25, Issue 6, 2021, Pages. 2075-2083
Received 25 April 2021; Accepted 08 May 2021.

RESULTS AND DISCUSSION

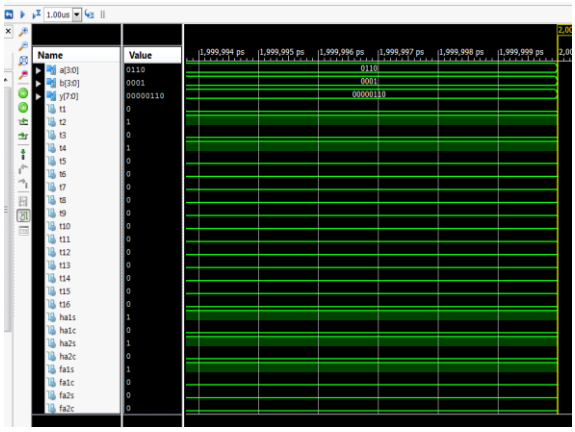

Figure 2 represents Output of the Product

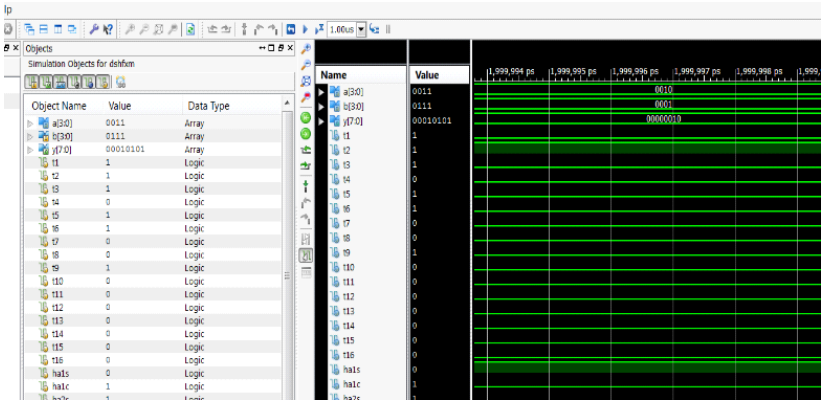

Figure 3 shows output of the stochastic multiplication.

Existing system:

Device Utilization Summary (estimated values)			[-]
Logic Utilization	**Used**	**Available**	**Utilization**
Number of Slices	78	960	8%
Number of 4 input LUTs	143	1920	7%
Number of bonded IOBs	46	66	69%

Table 1 represents the existing system

Annals of R.S.C.B., ISSN:1583-6258, Vol. 25, Issue 6, 2021, Pages. 2075-2083
Received 25 April 2021; Accepted 08 May 2021.

Proposed System:

Device Utilization Summary (estimated values)			[-]
Logic Utilization	**Used**	**Available**	**Utilization**
Number of Slices	68	960	7%
Number of 4 input LUTs	125	1920	6%
Number of bonded IOBs	45	66	68%

Table 2 represents the proposed system

The proposed has been simulated and the synthesis report can be acquired by utilizing Xilinx ISE 12.1i. The different parameters utilized for figuring existing and proposed system with Spartan-3 processor are given in the table.

Table 3:

s.no	Parameter	Existing	Proposed
1	Slices	78	68
2	LUT	143	125

Table 3 represents the comparision of proposed and existing system.

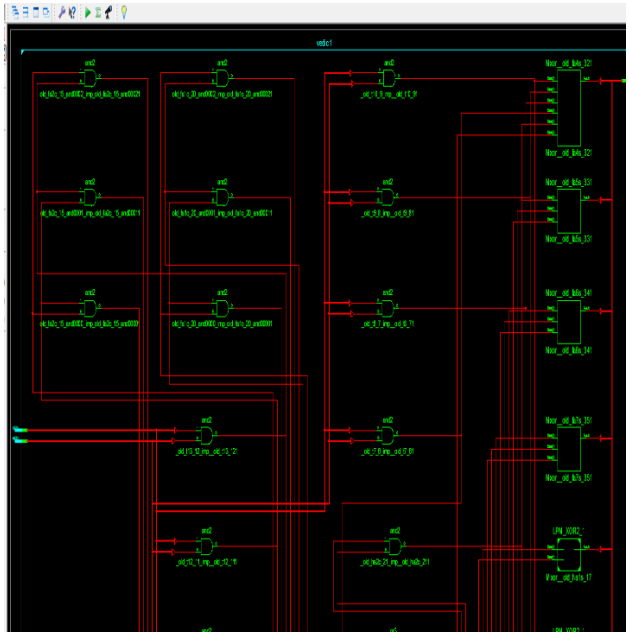

Figure 3 represents Simulation of Gatelevel Netlist

Annals of R.S.C.B., ISSN:1583-6258, Vol. 25, Issue 6, 2021, Pages. 2075-2083
Received 25 April 2021; Accepted 08 May 2021.

PERFORMANCE ANALYSIS

The diagram given below is shown that by using **Spartan 3** processor area and time will be reduced by implementing. By comparing the proposed system and existing system the area is reduced significantly.

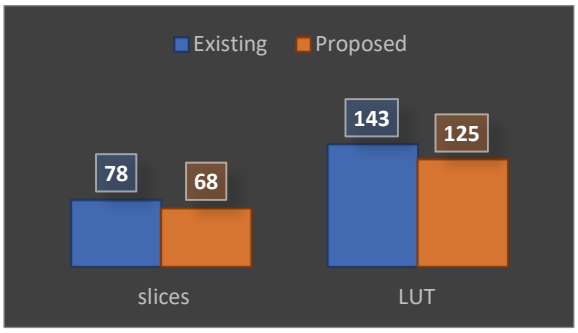

Figure 4 represents Comparision of proposed system and existing system.

CONCLUSION

Deep neural networks are state-of-the-art machine learning techniques for a good range of applications. Many of these applications are error-resilient, where approximate computing is naturally used to achieve great energy savings with minor quality degradation. In our project, Approximately Artificial neural network approximates the computation partial products in weight product stages, with the proposed solution, Approximately ANN achieves energy benefit with less than 5% quality loss for various multiplication techniques used in our experiments

REFERENCES

1) Bai, K., An, Q., Liu, L., & Yi, Y. (2019). A Training-Efficient Hybrid-Structured Deep Neural Network With Reconfigurable Memristive Synapses. IEEE Transactions on Very Large Scale Integration (VLSI) Systems, 1–14.

2) Choi, W., Duraisamy, K., Kim, R. G., Doppa, J. R., Pande, P. P., Marculescu, D., & Marculescu, R. (2018). On-Chip Communication Network for Efficient Training of Deep Convolutional Networks on Heterogeneous Manycore Systems. IEEE Transactions on Computers, 67(5), 672–686.

3) Dey, S., Nandi, S., & Trivedi, G. (2020). PowerPlanningDL: Reliability-Aware Framework for On-Chip Power Grid Design using Deep Learning. 2020 Design, Automation & Test in Europe Conference & Exhibition (DATE).

4) Edstrom, J., Das, H., Xu, Y., & Gong, N. (2019). Memory Optimization for Energy-Efficient Differentially Private Deep Learning. IEEE Transactions on Very Large Scale Integration (VLSI) Systems, 1–10.

Annals of R.S.C.B., ISSN:1583-6258, Vol. 25, Issue 6, 2021, Pages. 2075-2083
Received 25 April 2021; Accepted 08 May 2021.

5) Han, D., Lee, J., Lee, J., & Yoo, H.-J. (2019). A 1.32 TOPS/W Energy Efficient Deep Neural Network Learning Processor with Direct Feedback Alignment based Heterogeneous Core Architecture. 2019 Symposium on VLSI Circuits.

6) Luo, Y., Li, S., Sun, K., Renteria, R., & Choi, K. (2017). Implementation of deep learning neural network for real-time object recognition in OpenCL framework. 2017 International SoC Design Conference

7) Mody, M., Mathew, M., Jagannathan, S., Redfern, A., Jones, J., & Lorenzen, T. (2017). CNN inference: VLSI architecture for convolution layer for 1.2 TOPS. 2017 30th IEEE International System-on-Chip Conference (SOCC).

8) Sebastian, A., Boybat, I., Dazzi, M., Giannopoulos, I., Jonnalagadda, V., Joshi, V., … Eleftheriou, E. (2019). Computational memory-based inference and training of deep neural networks. 2019 Symposium on VLSI Technology.

9) Wang, M., Lu, S., Zhu, D., Lin, J., & Wang, Z. (2018). A High-Speed and Low-Complexity Architecture for Softmax Function in Deep Learning. 2018 IEEE Asia Pacific Conference on Circuits and Systems (APCCAS).

10) Yin, S., Jiang, Z., Kim, M., Gupta, T., Seok, M., & Seo, J.-S. (2019). Vesti: Energy-Efficient In-Memory Computing Accelerator for Deep Neural Networks. IEEE Transactions on Very Large Scale Integration (VLSI) Systems, 1–14.

11) Zhu, D., Lu, S., Wang, M., Lin, J., & Wang, Z. (2020). Efficient Precision-Adjustable Architecture for Softmax Function in Deep.