GreenTPU: Improving Timing Error Resilience of a Near-Threshold Tensor Processing Unit

Pramesh Pandey Prabal Basu Koushik Chakraborty Sanghamitra Roy
USU BRIDGE LAB, Electrical and Computer Engineering, Utah State University
{pandey.pramesh1, prabalb}@aggiemail.usu.edu,
{koushik.chakraborty, sanghamitra.roy}@usu.edu

ABSTRACT

The emergence of hardware accelerators has brought about several orders of magnitude improvement in the speed of the deep neural-network (DNN) inference. Among such DNN accelerators, Google Tensor Processing Unit (TPU) has transpired to be the best-in-class, offering more than 15× speedup over the contemporary GPUs. However, the rapid growth in several DNN workloads conspires to escalate the energy consumptions of the TPU-based data-centers. In order to restrict the energy consumption of TPUs, we propose GreenTPU—a low-power near-threshold (NTC) TPU design paradigm. To ensure a high inference accuracy at a low-voltage operation, GreenTPU identifies the patterns in the error-causing activation sequences in the systolic array, and prevents further timing errors from the same sequence by intermittently boosting the operating voltage of the specific multiplier-and-accumulator units in the TPU. Compared to a cutting-edge timing error mitigation technique for TPUs, GreenTPU enables **2X–3X** higher performance in an NTC TPU, with a minimal loss in the prediction accuracy.

1. INTRODUCTION

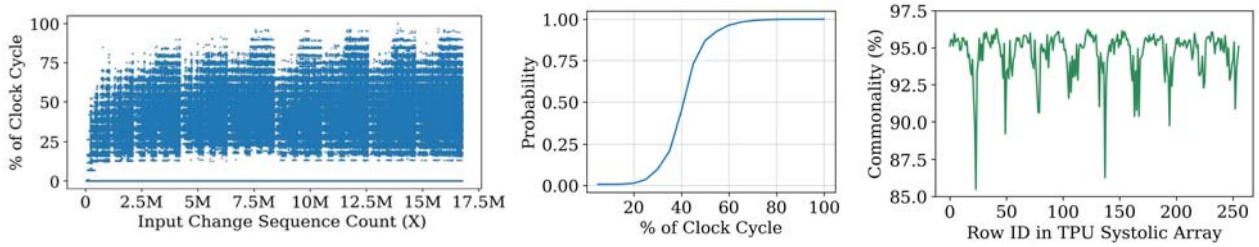
The cessation of Dennard's scaling, accompanied with the diminishing throughput from the growing number of on-chip cores, has led to the adoption of power-efficient domain-specific architectures. With the recent confluence of artificial intelligence (AI) and high performance computing, the domain-specific computing paradigm is already on the uprise, as evident by the success of the deep neural-network (DNN) accelerators [3,10,17]. Among the multitude of such ad-hoc AI architectures, the Google Tensor Processing Unit (TPU) is at the forefront, claiming 15× - 30× faster inference, compared to the top of the line CPUs and GPUs [7]. However, the unprecedented growth in the DNN workloads (e.g., speech recognition in Google Assistant [1,7]) portends a rapid increase in the overall power consumption of the Google data-centers. With a view to heavily curtailing the power consumption while sustaining a high inference accuracy, we envision a near-threshold (NTC) operation of the TPUs. However, operating a TPU at the NTC condition, can significantly dwindle the inference accuracy due to a high rate of timing errors [5,20]. This paper aims to exploit the inherent architectural artifacts of the TPUs, to predict and tackle the timing errors at NTC, thus promoting a reliable and energy-efficient low-power TPU design paradigm.

The high delay sensitivity to voltage and process variation (PV) at NTC necessitates a relaxed clock constraint to ensure an error-free execution. On the other hand, hardware accelerators like TPUs are designed to offer a high throughput in niche applications. So, in order to embrace the NTC design paradigm for TPUs, we need to adopt a better-than-worst case design strategy that can efficiently tolerate the timing errors in its systolic array architecture (Section 2.1). Prior research efforts delve into the challenges and solutions of tackling timing errors in conventional CPU and contemporary TPU architectures [6,20]. Next, we discuss why such existing techniques are not effective in an NTC TPU.

Razor—one of the most popular timing error detection and recovery schemes—employs a double sampling flip-flop to detect timing violations inside a pipeline stage [6]. The erring instruction is replayed at a reduced clock frequency to prevent a subsequent timing error. Adopting Razor in TPUs will negatively impact the performance, as the global timing error rate rapidly grows with the dimension of the systolic array. Hence, any recovery penalty, associated with correcting an erroneous computation, will significantly bloat the execution time of the inference. Zhang et al. have recently proposed TE-Drop—where an erring multiplier-and-accumulator (MAC) in a TPU, steals a clock cycle from its downstream MAC to correct the error, and bypasses the downstream MAC's update [20]. However, this approach cannot tackle any timing error in the last row of MACs, without incurring a significant performance penalty at NTC. As the partial sums grow towards the bottom of the systolic array, the impact of timing errors in the last row of MACs is the most crucial. Moreover, as the rate of the timing error increases significantly at NTC, bypassing the update of some MACs will greatly diminish the inference accuracy.

In the light of such shortcomings of the existing timing error mitigation techniques, we propose a novel *timing error prediction strategy*, exploiting the wavefront propagation of the data in a TPU systolic array. We observe that few activation sequences are more likely to cause timing violations in the MACs (Section 2.3). As the activation data streams through all the MACs in a row, an error-causing activation sequence, can serve as an excellent predictor to avoid subsequent errors in the rest of the MACs in the same row. We combine this early error prediction scheme with a low-complexity voltage boosting mechanism to propose GreenTPU—a new frontier in the design of reliable and low-

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from Permissions@acm.org.
DAC '19, June 2–6, 2019, Las Vegas, NV, USA
Copyright 2019 ACM. ISBN 978-1-4503-6725-7/19/06 ... $15.00
DOI: https://doi.org/10.1145/3316781.3317835.



(a) Exhaustive delay profile of a MAC unit.

(b) CDF of the delay profile.

(c) % of Commonality for all the 256 rows.

Figure 1: *Figure 1(a) shows the plot of the sensitization delays for all possible weights and input changes for a MAC unit. The variance in the input data can bring about ample delay variance. However, there are only few input sequences that can sensitize the longest delay paths, as depicted by the CDF plot in Figure 1(b). Figure 1(c) exhibits a very high % of Commonality (Equation 2) in the error causing input sequences for all the rows, during the inference of the MNIST dataset.*

power TPU. Following are the key contributions of our work:

- We observe that only few input data sequences cause timing violations in MACs. Consequently, they serve as an efficient predictor for impending timing errors (Section 2).
- We propose GreenTPU—a low-overhead NTC TPU design paradigm that predicts impending timing violations in the TPU systolic array, and precludes them using a novel voltage boosting mechanism (Section 3).
- Combining with our in-house statistical timing analyzer tool, we develop a TPU systolic array simulator in C++. We support an end-to-end integration, by interfacing our simulator with Keras [4], so as to closely emulate a real-life TPU-accelerated inference eco-system for contemporary DNN applications (Section 4).
- We demonstrate that GreenTPU provides two orders of magnitude reduction in timing errors at NTC, with respect to TE-Drop [20]—a cutting-edge timing error mitigation technique for TPUs (Section 5).
- Compared to TE-Drop, GreenTPU offers **2X–3X** higher performance in an NTC TPU, in 3 out of 4 DNN datasets, with only 3% average loss in the inference accuracy. Estimated from synthesis, place and route of a TPU systolic array RTL, augmented with GreenTPU, we find the area, power, and wire-length overheads to be ∼1.8%, ∼2.2%, and ∼4.1%, respectively (Section 5).

2. MOTIVATION

In this section, we demonstrate the opportunity of employing a predictive mechanism to tackle timing errors in NTC TPUs. Section 2.1 provides a background on the TPU systolic array. Using a cross-layer methodology (Section 2.2), we analyze the data-driven delay variance in the systolic array of MAC units (Section 2.3), and motivate the need for a timing error prediction scheme in NTC TPUs (Section 2.4).

2.1 Background

2.1.1 TPU Systolic Array

Matrix multiplication is the most expensive operation in the *inference* phase of the DNN applications. The usage of the systolic array of MAC units, has been recognized as a promising direction to accelerate the matrix multiplication. TPU—a DNN accelerator—employs a 256×256 systolic array of MAC units, to multiply the weight matrix with the activation (also referred to as *input*) matrix, maintaining a precision of 8-bit integer [7]. The weights are pre-loaded into the MACs. The activations stream from the left to the right columns of the array at successive clock cycles. The partial sums from the rows of MACs move downstream. Unlike CPUs and GPUs, *a TPU boasts a distinctly homogeneous architecture with a highly predictable data-flow pattern.*

2.1.2 Hazards and Opportunities of NTC TPUs

Operating a TPU at the NTC condition ideally contributes to a quadratic saving in the energy consumption. However, the performance of the TPU heavily declines due to a large delay experienced by the circuits at an NTC voltage [5]. Moreover, a high delay sensitivity to PV and voltage variation at NTC, demands the clock frequency to be heavily relaxed, compared to a super-threshold operation. Hence, in order to operate with an aggressive clock constraint at NTC, a TPU needs to efficiently tolerate a high rate of timing violations. Furthermore, due to a very deep pipelined architecture of the systolic array (Section 2.1.1), even a small rate of timing error aids to a severe drop in the inference accuracy of the DNN applications [20].

Fortunately, the architectural homogeneity and a predictable data-flow pattern in TPUs, offer a unique opportunity to efficiently tackle timing errors at NTC. Owing to a fixed 8-bit precision in the arithmetic operations, we get a finite state space of different sensitized path delays, experienced by the MAC units. *Isolating the subset of the relatively high delays, and correlating that subset with the concerned data patterns, can facilitate the prediction of the impending timing errors in the TPU systolic array.*

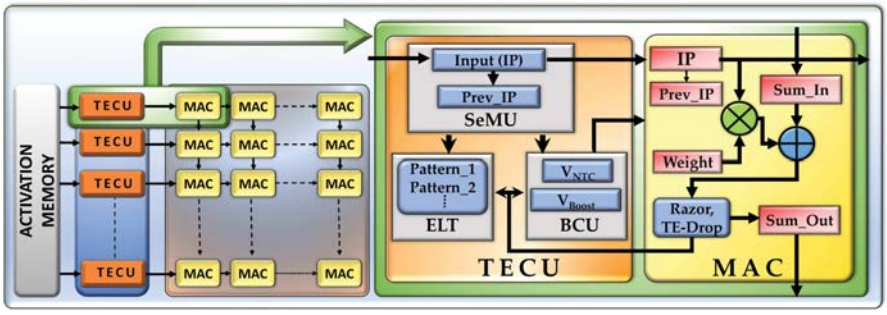
2.2 Methodology

We synthesize a MAC unit at an NTC operating condition (Section 5), by using the 15-nm FinFET library from NanGate [16]. We employ our in-house statistical timing analysis tool to study the delay distributions of the sensitized paths for different inputs to the MAC unit. For a conservative estimate, we consider PV-induced delays, obtained from VARIUS-NTV [18], in randomly chosen 2% of the gates in the MAC circuit [19]. We further elaborate on our cross-layer methodology in Section 4.

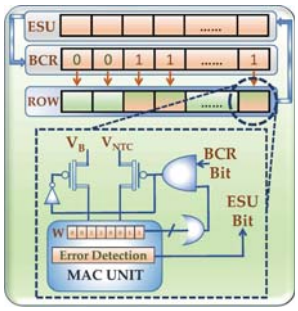
2.3 Results and Significance

The multiplier block of a MAC unit has a relatively deeper logic depth, compared to the accumulator. Hence, we model the delay distribution of the MAC, as a function of the change in inputs to the multiplier, i.e., the activation sequence, and the weight. We create an exhaustive set of 8-bit activation sequences for all possible 8-bit weights, leading to a total of 16,777,216 unique combinations.

Figure 1(a) shows the delay profile of a PV-affected MAC unit at NTC, obtained by providing all the aforementioned combinations of weights and input changes. A value of X in



(a) A conceptual block diagram of GreenTPU.



(b) MACs-BCU interaction.

Figure 2: *Figure 2(a) shows that the TECUs are pipelined between the activation memory and the rows of the systolic array of MACs. A timing error inside a MAC unit is detected and tackled using Razor and TE-Drop techniques, respectively. A TECU comprises an ELT, an SeMU, and a BCU. ELT stores the error-causing input patterns. SeMU, on the other hand, monitors the input data stream and queries the ELT, to identify potential error-causing input sequences. The BCU (Figure 2(b)), comprising two 256-bit registers—ESU and BCR—prevents future timing errors by boosting the operating voltage of the MACs in a row.*

Figure 1(a), corresponds to a specific input change sequence, for a specific weight W, as expressed in equation 1.

$$\begin{aligned} Weight\,(W) &= \lfloor \frac{X}{65536} \rfloor,\ S = X \bmod 65536 \\ Input\ change &: \lfloor \frac{S}{256} \rfloor \rightarrow (S \bmod 256) \end{aligned} \tag{1}$$

The delay profile shows ample variation, resulting from the variance in the input data. This delay variation is statistically shown as a CDF plot of the delay values in Figure 1(b), where we conservatively attribute the maximum delay to be the clock period. The key observations from Figure 1(a) and 1(b) are: (a) no paths are sensitized when the same activation sequence is applied in two consecutive cycles, and (b) a majority of the multiplication operations sensitize paths with low delays. For instance, we notice that the set of delays with more than 60% of the clock cycle is only 3.6% of the entire state space of delays. This sparse sensitization of the higher delay paths, eases the prediction of the recurring timing errors from the same input sequence. Next, we discuss the insight to our proposed design of GreenTPU.

2.4 Timing Error Prediction in TPUs

We aim to systematically study the likelihood of an error-causing input sequence in a MAC, to produce timing errors in the subsequent MACs, belonging to the same row. In this pursuit, we propose a *Commonality* metric in Equation 2.

$$Commonality_i(\%) = 100 * \left(1 - \frac{\bigcup_{j=0}^{255} UES_j}{\sum_{j=0}^{255} UES_j}\right) \tag{2}$$

where, UES_j is the set of unique input sequences that cause timing errors in the j^{th} MAC unit of the i^{th} row.

Figure 1(c) shows a plot of the *Commonality*(%), measured across all the 256 rows during the inference of 1000 test inputs of the MNIST dataset. We observe that, for all the rows, the commonality of the error-causing input sequences is more than 85%. This result indicates a *landslide effect* of timing errors in the systolic array of a TPU. In other words, if an input sequence causes a timing error in a MAC unit, that sequence is very likely to cause timing errors in the subsequent MACs, until the sequence is alive in the row. Hence, predicting errors based on the input sequences, and adopting a row-wise control strategy can greatly reduce the number of timing errors in a TPU. With this insight, we next discuss GreenTPU—our proposed energy-efficient TPU systolic array design, for a near-threshold operation.

3. GREENTPU

GreenTPU is a novel low-power TPU design paradigm, that dynamically predicts and tackles timing errors in the systolic array of MAC units. Section 3.1 outlines the design overview. The details of the components of GreenTPU are elaborated in Section 3.2 through 3.4.

3.1 Design Overview

Figure 2(a) depicts the top-level design overview of GreenTPU. The heart of GreenTPU is the Timing Error Control Unit (TECU). TECU is responsible for predicting and preventing timing errors in the MAC units. In order to maintain a low-complexity circuit design while incurring a negligible performance overhead, we dedicate one TECU per row of MACs, pipelined between the activation memory and the systolic array. A TECU has three main components, viz., Error Log Table (ELT), Sequence Monitor Unit (SeMU), and Boost Control Unit (BCU). When a timing error occurs in any MAC unit of a row, the ELT logs the timing error causing input sequence pattern. Simultaneously, the BCU is alerted to boost the operating voltage of the subsequent MACs in the row, in order to prevent any future timing error. The SeMU monitors the sequence of inputs, and tries to find a matching pattern in the ELT in every clock cycle. If a match is found, SeMU communicates with the BCU to preclude future timing errors in all the MAC units of a row.

3.2 Error Log Table (ELT)

ELT is a look-up table which stores the patterns of the input sequence that lead to timing errors in a MAC unit. We observe that different bits of an input sequence have different degrees of contribution to the sensitization delay. As a consequence, many different input sequences can be clubbed to fewer representative patterns in a space-efficient manner. Algorithm 1 shows our heuristic to represent an input sequence as a specific pattern. The pattern is simply a bit-wise XOR of the inputs in the sequence, which captures the information on the specific bits with different contributions to the sensitization delay. For any new input sequence, the XOR-based pattern is computed and compared with the stored patterns, based on an empirically determined *pattern_match_threshold* (line 3 to 11 in Algorithm 1). If no match is found, then the new pattern is stored in the table.

We augment each MAC unit with the capability to store

Algorithm 1 Pattern Matching Heuristic

```
1: TH ← pattern_match_threshold
2: procedure MATCH(current_activ, previous_activ)
3:     new_pattern ← current_activ ⊕ previous_activ
4:     for all saved_pat ∈ saved_patterns do
5:         similarity ← saved_pat | new_pattern
6:         num_zeros_sim ← num_reset_bits(saved_pat)
7:         num_zeros_new ← num_reset_bits(similarity)
8:         if num_zeros_new > ⌊TH × num_zeros_sim⌋ then
9:             return match_found
10:        end if
11:    end for
12:    save(new_pattern)
13: end procedure
```

the previous clock cycle's activation input, thus enabling it to infer the input sequence. A timing error in each MAC unit is sensed using a double-sampling flip-flop at the output, similar to Razor [6]. We prevent an erroneous computation from the timing error by employing TE-Drop [20], where the errant MAC steals a clock cycle from its downstream MAC to correctly finish its own update. An 8-bit pattern corresponding to this error causing input sequence is sent by the MAC unit as a new entry to the ELT, while the correct output is being computed parallelly. Also, the BCU is signalled with the errant MAC unit's position in the row, to prevent further timing errors in the MAC units, located to the right of the errant MAC. The ELT is implemented as a content addressable memory that enables a fast lookup. When the ELT is full, a pseudo-LRU-based eviction policy is used (not shown in Algorithm 1) to replace an existing pattern with the new incoming pattern. The size of the ELT is a trade-off between the hardware overhead and prediction accuracy, which is discussed in Section 5.

3.3 Sequence Monitor Unit (SeMU)

SeMU identifies the possibility of a recurring timing error. The input activation data, coming to each row, is intercepted by SeMU, as the TECU is placed in pipeline between the activation memory and the systolic array. For a given activation sequence coming from the activation memory, SeMU checks if a corresponding pattern is already present in the ELT, based on Algorithm 1. If a match is found, the BCU is alerted to boost the operating voltage of some of the MACs in the row (Section 3.4). This action is taken in order to prevent the timing errors that would have been caused by the input sequence. Due to its pipelined architecture, SeMU adds a negligible performance overhead.

3.4 Boost Control Unit (BCU)

BCU is responsible for boosting the operating voltage of the MAC units, in order to prevent timing errors. As shown in the Figure 2(b), a BCU houses two 256-bit registers: Boost Control Register (BCR) and Error Sensing Unit (ECU). Each bit of these registers corresponds to each MAC unit in a row. We adopt the boosting technique proposed in [15], where every MAC unit has access to two voltage rails, V_{NTC} and V_B, representing a near-threshold and a boost voltage, respectively. The reset (set) value in any bit of the BCR, indicates the corresponding MAC unit to operate with the V_{NTC} (V_B) voltage. In our experiments, we set V_{NTC} and V_B to 0.45V and 0.65V, respectively. Employing the transition infrastructure of [15], we notice that the switching between V_{NTC} and V_B can be performed within one clock cycle of the NTC TPU.

We observe that if the pre-loaded weight of a MAC unit is zero, it is unlikely to encounter a timing error. Hence, a MAC with weight zero can disable the voltage boost for itself, to conserve energy. Whenever a timing error occurs in any MAC unit, the corresponding bit in the ESU register is set. Based on the position of this set bit, a certain number of bits in the BCR, that are located to the right of that position, are periodically set. As a result, the MAC units, specific to those set bits in the BCR, will be boosted in the subsequent cycles, precluding any probable timing violations. The number of bits in the BCR to be set, is determined empirically, so as to trade-off between the tolerable timing errors, and the energy overhead due to boosting.

If an errant pattern is found by the SeMU (Section 3.3), BCU periodically sets a certain number of bits of the BCR in succeeding epochs. Once again, the number of bits to set, as well as the size of the epoch, are empirically ascertained. These choices are guided by the energy budget of the GreenTPU implementation, and the noise margin of the TPU systolic array at NTC.

4. METHODOLOGY

In this section, we explain our extensive cross-layer methodology, used to implement and evaluate GreenTPU.

4.1 Device Layer

We estimate the NTC energy consumptions by performing HSPICE simulations on the basic logic gates (viz., NAND, NOR and Inverter). We use the 31-stage FO4 inverter-chain as a representative of various combinational logics in a TPU. The simulation parameters are obtained from the 16-nm Predictive Technology Model [21]. We incorporate the impact of the PV at NTC using the VARIUS-NTV [8] model. The FinFET characteristics are obtained using the VARIUS-TC model [9]. The delays of the basic gates are used in the circuit layer (Section 4.2) to ascertain the sensitized path delays in a MAC.

4.2 Circuit Layer

We develop the Verilog RTL description of a systolic array, and augment it with the GreenTPU components. We synthesize the RTLs using the Synopsys Design Compiler, at various operating conditions. We perform place and route of the synthesized netlist using Cadence SoC Encounter, and estimate the area, power, and wirelength overheads at the NTC operating condition. Using both synthetically generated, as well as, real dataset driven inputs, we obtain the sensitized path delays in the MAC array with our in-house statistical timing analysis (STA) tool. Based on a library of the delay files of the basic logic gates at different operating voltages, the STA tool reports the delays of the sensitized paths of the MAC circuit.

4.3 Architecture Layer

Based on the architectural description detailed in [7], we develop a cycle-accurate TPU systolic array simulator—TPU-Sim—in C++, and implement the GreenTPU components in TPU-Sim. We integrate the STA tool (Section 4.2) with TPU-Sim, to accurately model timing errors in the MACs, based on real data-driven sensitized path delays. We create a real TPU-based inference eco-system by conjoining TPU-Sim with Keras [4]. First, we train several DNN applications (viz., MNIST [12], Reuters [2], CIFAR-10 [11],

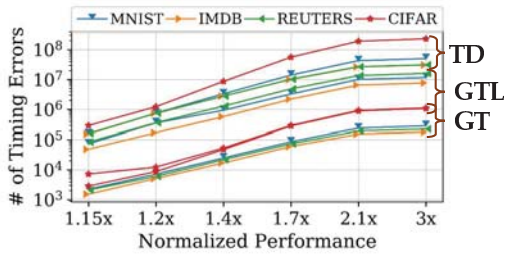


Figure 3: *Number of timing errors encountered in different comparative schemes across 4 DNN datasets.*

IMDB [14]) using Keras, running TensorFlow in the backend. We extract each layer's activation inputs and trained model weights, and pre-process them into multiple 256×256 8-bit-integer matrices. TPU-Sim is invoked with each pair of the pre-processed input and weight matrices. The output matrices from the TPU-Sim are combined to evaluate the inference accuracy. We parallelize our framework for handling numerous test data using Python Multiprocessing.

5. EXPERIMENTAL RESULTS

In this section, we evaluate the efficacy of different timing error-resilient schemes, when a TPU operates at a better-than-worst-case scenario. Our baseline NTC operating condition (0.45V, 67MHz) guarantees an error-free execution of the TPU. Section 5.1 describes the comparative schemes. Section 5.2 elaborates the timing error resilience of the schemes. Section 5.3 presents the inference accuracy and the energy consumption of the TPU under different schemes. Finally, Section 5.4 discusses the hardware overheads of GreenTPU.

5.1 Comparative Schemes

- **TE-Drop (TD):** This is a recently proposed technique that can tackle timing errors in the systolic array of a TPU [20]. The errant MAC steals the next clock cycle from its downstream MAC to correct the error, while the downstream MAC bypasses its own operation.
- **GreenTPU (GT):** This is our proposed design strategy that stores the error-causing patterns in order to predict any imminent timing errors from those patterns (Section 3). The imminent errors are precluded by boosting the operating voltage of the rows of MACs. GT employs the pattern matching heuristic (i.e., Algorithm 1), with a pattern-match threshold of 90%, and an ELT size of 10.
- **GreenTPU-Lite (GTL):** This is a variant of GreenTPU that stores only one error-causing pattern. The design complexity is significantly reduced by replacing the ELT with an 8-bit register. GTL utilizes Algorithm 1, with the pattern-match threshold set to 100%, thereby triggering the boosting mechanism only upon an exact match between the incoming pattern and the stored pattern.

5.2 Timing Error Resilience

Figure 3 depicts the number of timing errors encountered during the inference of the DNN datasets under different schemes, when the TPU operates at a higher frequency, compared to the baseline. However, at all frequency—denoted by the X-axis—the operating voltage is kept constant at 0.45V. The Y-axis values are represented in a logarithmic scale. We notice that, *on an average, GT encounters two orders of magnitude less timing errors, with respect to TD, across all the datasets, at any higher performance level.* This is due to the fact that unlike TD, GT can predict imminent timing errors and prevent them from occurring. GTL, despite being significantly more error-resilient than TD, incurs more timing errors than GT. The storage of only one error-causing pattern substantially dwarfs the prediction mechanism in GTL. For CIFAR-10, GT cannot outplay GTL at higher performance points. This anomaly is attributed to the extreme variance in the activation patterns of CIFAR-10. Such a variation engenders frequent ELT evictions while vastly widening the intervals of recurrent errors from the same patterns. Consequently, even a moderate size of the ELT is ineffective in tackling the impending timing errors.

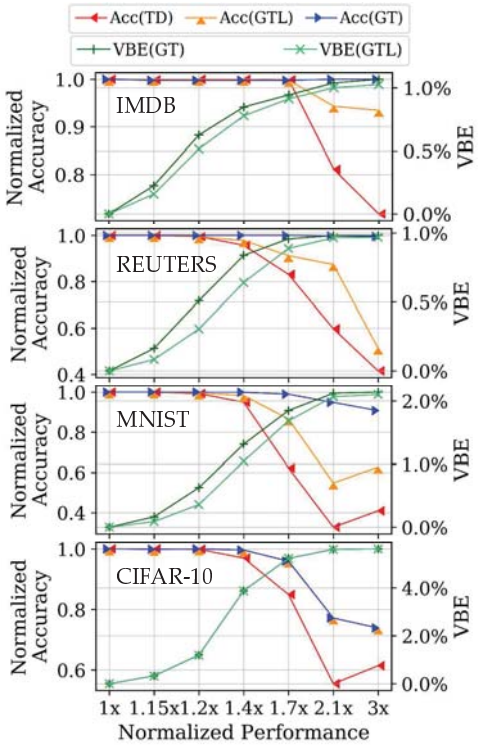


Figure 4: *Normalized inference accuracy (Acc) from the comparative schemes, and the voltage boost energy (VBE) consumptions of GreenTPU and GreenTPU-Lite, at different normalized performance levels, across 4 DNN datasets.*

5.3 Inference Accuracy and Energy

Figure 4 presents the variations in the inference accuracy at different performance points (Section 5.2), under various comparative schemes (Section 5.1), for 4 DNN datasets. The accuracy values of the datasets are normalized to the corresponding error-free accuracy (IMDB: 0.90, CIFAR-10: 0.77, MNIST: 0.98, REUTERS: 0.80) from the baseline NTC TPU. Figure 4 also shows the voltage boosting energy (VBE), associated with the boosting mechanism in GT and GTL. VBE is calculated as a percentage of the energy consumption of the baseline NTC systolic array with no augmentation.

Up to 1.4× the baseline performance, all the schemes can efficiently prevent the impact of timing errors from affecting the inference accuracy. However, as the performance is further increased, GT and GTL offer considerably better accuracies with respect to TD, for all the datasets. This is due to the high timing error resilience of GT and GTL (Section 5.2). The pattern matching capability, along with

a larger ELT, makes GT a more effective scheme, compared to GTL. *Our baseline NTC TPU, augmented with GT, can be operated at 2×–3× the baseline frequency, with only 3% average loss in the inference accuracy for 3 out of 4 DNN datasets.* For CIFAR-10, GT is only as effective as GTL, which is consistent to their similar timing error resilience in this dataset (Section 5.2).

The VBE of GTL—owing to its lower hardware footprint and infrequent boosting—is usually less than the VBE of GT. However, for CIFAR-10, both the schemes trigger the boosting mechanism for the same number of time, thus incurring similar energy overheads. Despite a monotonic increase with the performance, VBEs of our proposed schemes are limited to ∼6% of the baseline NTC TPU energy consumption. This result is due to the sporadic occurrence of the boosting. *Hence, GreenTPU serves as an extremely error-resilient and energy-efficient design paradigm that can unlock a high performance in future low-power NTC TPUs.*

5.4 Implementation Overheads

The hardware overheads of GreenTPU come from the TECU components, the additional voltage rail, and the augmentation of each MAC with the Razor capability. The area overhead of GreenTPU is estimated to be ∼1.8%. This small footprint is attributed to the fact that the systolic array occupies only 24% of the overall TPU die area [7]. GreenTPU incurs a power overhead of ∼2.2%, compared to the vectorless power consumption of the systolic array. From the detailed route reports, GreenTPU's wire-length overhead is estimated to be ∼4.1%.

6. RELATED WORK

Prior research efforts related to our work deal with improving the energy efficiency of the DNN hardware accelerators through various means. Chen et al. proposed an optimal MAC operation mapping rule, called Row-Stationary dataflow, that optimizes the data movement inside a deep convolutional neural network (CNN), resulting in a superior system-level energy efficiency [3]. Reagen et al. demonstrated an automated co-design approach across the algorithm, architecture, and circuit to offer a staggering 8.1× power reduction over a baseline DNN accelerator, without compromising the accuracy [17]. Lin et al. presented a statistical error compensation technique to correct process variation induced timing errors in CNNs, operating under near-threshold condition [13]. Zhang et al. proposed a timing speculation approach that enables an aggressive voltage underscaling in DNN accelerators without compromising the classification accuracy [20]. Kim et al. presented a memory adaptive training with in-situ canaries, that enables aggressive voltage scaling of DNN-accelerator weight memories to improve the energy-efficiency [10]. *However, our work is the first one that exploits the data-driven delay variance in the systolic array of MACs, to predict timing errors in TPUs, operating under near-threshold condition.*

7. CONCLUSION

The unprecedented growth of the DNN workloads in the recent years, requires an energy-efficient DNN accelerator design paradigm, that can offer an optimal inference accuracy at a high performance. In this paper, we present GreenTPU—an energy-optimized systolic array design for Google TPU—a state-of-the-art DNN accelerator. Operating at the NTC condition, GreenTPU can efficiently predict and prevent the imminent timing errors in its systolic array of MACs, thus offering close to an error-free accuracy with a high performance. Compared to a recently proposed timing error mitigation strategy for TPUs, GreenTPU enables 2×–3× higher performance in an NTC TPU, with a minimal loss in the prediction accuracy, and minor hardware footprints.

Acknowledgements

This work was supported in part by National Science Foundation grants (CAREER-1253024, CCF-1318826, CNS-1421022, and CNS-1421068). Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

8. REFERENCES

[1] OK Google, Siri, Alexa, Cortana; Can you tell me some stats on voice search? `https://edit.co.uk/blog/google-voice-search-stats-growth-trends/`.

[2] Reuters-21578 Dataset.

[3] Chen, Y.-H. and others Using dataflow to optimize energy efficiency of deep neural network accelerators. *IEEE Micro 37*, 3 (2017), 12–21.

[4] Chollet, F., et al. Keras. `https://keras.io`, 2015.

[5] Dreslinski, R. G. and others Near-Threshold Computing: Reclaiming Moore's Law Through Energy Efficient Integrated Circuits. *Proc. of the IEEE 98*, 2 (2010), 253–266.

[6] Ernst, D. and others Razor: A Low-Power Pipeline Based on Circuit-Level Timing Speculation. In *Proc. of MICRO* (2003), pp. 7–18.

[7] Jouppi, N. P. and others In-datacenter performance analysis of a tensor processing unit. In *Computer Architecture (ISCA), 2017 ACM/IEEE 44th Annual International Symposium on* (2017), IEEE, pp. 1–12.

[8] Karpuzcu, U. R. and others VARIUS-NTV: A microarchitectural model to capture the increased sensitivity of manycores to process variations at near-threshold voltages. In *DSN* (2012), pp. 1–11.

[9] Khatamifard, S. K. and others VARIUS-TC: A modular architecture-level model of parametric variation for thin-channel switches. In *ICCD* (2016), pp. 654–661.

[10] Kim, S. and others Energy-Efficient Neural Network Acceleration in the Presence of Bit-Level Memory Errors. *IEEE Transactions on Circuits and Systems I: Regular Papers*, 99 (2018), 1–14.

[11] Krizhevsky, A. Learning Multiple Layers of Features from Tiny Images.

[12] LeCun, Y., and Cortes, C. MNIST handwritten digit database.

[13] Lin, Y. and others Variation-tolerant architectures for convolutional neural networks in the near threshold voltage regime. In *Signal Processing Systems (SiPS), 2016 IEEE International Workshop on* (2016), IEEE, pp. 17–22.

[14] Maas, A. L. and others Learning Word Vectors for Sentiment Analysis. Association for Computational Linguistics, pp. 142–150.

[15] Miller, T. N. and others Booster: Reactive Core Acceleration for Mitigating the Effects of Process Variation and Application Imbalance in Low-Voltage Chips. In *HPCA* (2012), pp. 1–12.

[16] NanGate. `http://www.nangate.com/?page_id=2328`.

[17] Reagen, B. and others Minerva: Enabling low-power, highly-accurate deep neural network accelerators. In *ACM SIGARCH Computer Architecture News* (2016), vol. 44, IEEE Press, pp. 267–278.

[18] Sarangi, S. and others VARIUS:A Model of Process Variation and Resulting Timing Errors for Microarchitects. *IEEE Tran. on Semicond. Manufac. 21* (2008), 3 –13.

[19] Shabanian, T. and others ACE-GPU: Tackling Choke Point Induced Performance Bottlenecks in a Near-Threshold Computing GPU. In *ISLPED* (2018).

[20] Zhang, J. and others ThUnderVolt: Enabling Aggressive Voltage Underscaling and Timing Error Resilience for Energy Efficient Deep Neural Network Accelerators. *arXiv preprint arXiv:1802.03806* (2018).

[21] Zhao, W., and Cao, Y. New Generation of Predictive Technology Model for sub-45nm Early Design Exploration. *T. Electron Devices 53*, 11 (2006), 2816 –2823.