RETROSPECTIVE: Cnvlutin: Ineffectual-Neuron-Free Deep Neural Network Computing

Jorge Albericio[1] Patrick Judd[1] Tayler Hetherington[2] Tor Aamodt[3] Natalie Enright Jerger[4] Andreas Moshovos[4]
[1]NVIDIA [2]Oracle [3]University of British Columbia [4]University of Toronto

In the late 2000s/early 2010s, we, as others in the architecture community, were transitioning from power-aware strategies for general-purpose computing to accelerators. We were somewhat familiar with neural networks, e.g., while a senior in 1996, Tor completed Geoff Hinton's grad course and an RL-based thesis, whereas Patrick developed an interest in machine learning and computer vision while writing his master's thesis. In 2009, Andreas, prompted by Jonathan Rose, established the first CUDA programming course in Toronto, building over materials from Wen-Mei Hwu and David Dirk. Alex Krizhevsky's class project, a convolutional neural net, led to a meeting with Geoff Hinton. The success of neural networks was uncertain, but we found the prospect of accelerating them with reconfigurable hardware or custom GPU kernels intriguing.

Meanwhile, other collaborative research efforts began to materialize, and the flexibility they afforded enabled us to set our own objectives and helped us have Jorge, a recent Ph.D. graduate from the University of Zaragoza, join us in 2013. Around this time, the excitement surrounding deep learning had intensified, particularly due to our close proximity to the Computer Sciences department. Raquel Urtasun, who has since become a prominent figure in machine learning and its applications, particularly in autonomous driving, joined Toronto at the time. We were fortunate to establish an early collaboration with her. Jorge dedicated himself to learning about machine learning, frequented Raquel's course and met with her group several times. It took significant time and effort to learn to "speak each other's language".

As we ventured into machine learning acceleration, we leaned on our collective experience with architectural optimizations for general-purpose systems where many optimizations, such as caches and branch predictors, exploit naturally emerging behaviors in applications. It was clear to us that several "front-line" optimizations, such as data blocking and parallelism, would receive immediate attention. These *computation-structure-based* techniques have proven effective in other application domains yet applying them effectively is a non-trivial task. We assumed that many would focus on these optimizations first given their potential for immediate and tangible benefits. As academics, we believed our efforts would be more valuable in identifying techniques that *could* prove useful in the future. Consequently, we chose to pursue *behavior-based* optimizations. Our aim was to examine deep learning models and pinpoint naturally emerging behaviors that could help us automatically reduce computations and data transfers. We assumed that neural network architectures will evolve, thus to increase the chances of staying relevant, we concentrated on elemental operations and data transfers. Like many other groups at the time, we focused on convolutional neural networks which were the most mature. Finally, we assumed that as in general-purpose computing, it would be best not to *require* that the developers modify the models to best fit the hardware we had in mind. Whatever technique we were to develop had to work with *unmodified models* and where possible to reward hardware-aware model optimizations.

Patrick was inspired by Joshua San Miguel's and Natalie's work on approximate computing to apply similar principles to machine learning acceleration. Early on, he noted that the datatypes utilized by these models could be further optimized. He devised a heuristic, profile-based method for post-training quantization to minimize per tensor bit widths. This insight led to two precursor works to Cnvlutin that fed into it. The quantization method developed in collaboration with Raquel, was submitted to ICLR but failed to convince of its importance. It was eventually published as a journal paper. Proteus was a hardware/software co-design that utilized per tensor bit widths to minimize footprint. A key concept in Proteus was Patrick's idea to pack variable length encoded values into virtual columns to reduce lateral movement avoiding wide crossbars. After a few iterations, it was published at ICS.

Meanwhile, Jorge was analyzing neural models with a specific aim to observe value properties and other emerging behaviors. He discovered that many filters had similar or even identical weights at corresponding positions. We hypothesized this might be due partly to how the models function and partly to the filters' cuboid shape having to encapsulate complex-shaped features. He also noticed that many values, especially on the activation side, were zeros. Intriguingly, while the zero activations would change with the input, their overall frequency remained relatively constant. We hypothesized these represented locations or features deemed non-important or absent and that ReLU clipped to zero. Focusing on the first layer in image classification tasks gave us an opportunity to rationalize this phenomenon.

These two value behaviors led to Convoy, an attempt at an accelerator design that predated Cnvlutin that Jorge led and pursued with Patrick and Siu Pak Mok, a student of our beloved colleague Greg Steffan that left us too early. Convoy initially aimed at reusing computations across filters and possibly other dimensions. It morphed into an effort to also take advantage of the zeros by removing computation. While ultimately Convoy never materialized it served as an excellent learning experience helping us appreciate the challenges of energy efficiency, especially when data movement is involved. Convoy also introduced concepts that we later revisited and refined, leading to solutions for extracting sparsity.

In Cnvlutin, our aim was to eliminate zero-involved computations and, if possible, data movement. We focused on activations, considering them more challenging and therefore more interesting to address. Our design, refined over numerous in-depth meetings, was based on DaDianNao's processing element [1] which exploited spatial and temporal reuse. The key challenge in Cnvlutin was minimizing data movement and expensive crossbars. We chose to support only temporal movement to avoid the high cost of arbitrary spatial and temporal movement. This reflects a fundamental trade-off in computer architecture: it's not necessary to leverage every opportunity, just enough to provide significant benefits. We realized during Cnvlutin's development that there's ample time to rearrange layer outputs before they become inputs for the next layer. This enabled Cnvlutin to organize values in memory so that the non-zero ones appeared in sequence when read back. Furthermore, drawing on Patrick's previous quantization work, we found many near-zero values could be treated as zero, amplifying benefits and enabling a trade-off between energy efficiency and accuracy.

Evaluating Cnvlutin necessitated the development of a new methodology, incorporating existing concepts and tools. We created an in-house simulator for cycle measurement and used established tools for memory energy and area modeling. Tayler's expertise was invaluable in modeling the energy and area of the custom logic. To this day, accessing technology nodes superior to 65nm remains a challenge for Canadian researchers. We also "opted" not to build test chips for Cnvlutin or our other early projects. Building a chip remains prohibitively expensive given our limited resources and time afforded with our students. We opted instead to explore additional techniques.

We're grateful to the ISCA'16 reviewers who appreciated our work and provided supportive, constructive feedback. We feel honored to have our work featured alongside several influential papers on ML acceleration at the same event. The appreciation shown tremendously boosted our motivation and eagerness to continue our work. Cnvlutin inspired us to delve further into ML acceleration and set the groundwork for broader initiatives in Canada. Our subsequent research has encompassed bit sparsity for inference and training, advanced post-training quantization approaches, sparsity extraction, efficient datatype learning, lossless and lossy compression methods, data fetching policies, reducing computations and data footprint during training while provably maintaining accuracy, and resilience to attacks. Would we implement all these techniques in an ML accelerator design? Certainly some we would, but definitely not all. Yet, we don't regret exploring any of them. We value understanding under which conditions each technique is viable and of having inspired follow up work.

Collaboration and funding are crucial to any successful program. Natalie, Tor, and Andreas initiated an effort to establish a Canada-wide research network on ML acceleration. Despite initial mixed reactions, we ultimately formed the NSERC COHESA research network, thanks to local industry support. The consortium of 23 colleagues and 5 companies focuses on the exploration of hardware and software acceleration techniques. Although the 6-year program is now concluding, its impact will persist, both through the work it facilitated and the young researchers it mentored. Securing sufficient support for architecture work and fostering appreciation or understanding for its importance continues to present challenges. Establishing and sustaining a research team takes years of effort, but it can be disrupted in a matter of months.

In terms of publication, besides Cnvlutin and Stripes, gaining acceptance for follow up work has grown increasingly arduous, discouraging younger researchers and impeding their career and work advancement. This is a challenge that extends beyond the realm of computer architecture. As a community, we often hinder our own progress by placing excessive emphasis on achieving "perfection" rather than evaluating the value of advancements in research. This particularly challenging in machine learning acceleration that is receiving universal attention in the broad research community. Exploring alternatives and establishing when they are preferable is a foremost goal of research. It is not about proving that a technique is universally better under all possible perceived conditions/goals. It is about feeding into further innovation and informing practical designs regarding the suitability of alternatives.

After leaving Toronto, Jorge joined NVIDIA, continuing his work on deep learning acceleration. He then joined Cerebras, contributing to the architecture of their unique technology, before returning to NVIDIA. There, he applies a holistic perspective on deep learning and related applications, informed by his initial, elemental operation-level work and understanding of hardware-level implications. He and his NVIDIA colleagues devised the industry-leading structural sparsity solution. After graduation, Patrick also joined NVIDIA, focusing on enhancing deep learning support and has been a major contributor to the design of efficient data types for training and inference. After UBC, Tayler joined Oracle, where he worked on developing methods for interpreting the behavior of machine learning models, and has since transitioned to building a high-performance cloud database service.

References

[1] Y. Chen, T. Luo, S. Liu, S. Zhang, L. He, J. Wang, L. Li, T. Chen, Z. Xu, N. Sun, and O. Temam, "DaDianNao: A Machine-Learning Supercomputer," in *Microarchitecture (MICRO), 2014 47th Annual IEEE/ACM International Symposium on*, Dec 2014, pp. 609–622.