yJackson Fries
3/3/2024
Block 2: Hardware Speedup

Why Do We Want Dedicated ML Hardware

One of the leading ways machine learning is being advanced right now is through the use of specialized hardware whether it's in the form of larger models with NVIDIA's focus on data centers to smaller more use specific machine learning on a system to system basis with FPGAs and new dedicated machine learning chips on modern GPUs. The reason why there is so much focus on dedicated hardware for machine learning is that as AI and machine learning becomes more and more prevalent in modern times having dedicated hardware for it will greatly increase the performance in all aspects from training time, classification time and more. There are a couple different types of machine learning hardware that are used today, some of the most popular ones include CPUs,GPUs, FPGAs and ASICs. Some of these different methods are super application specific such as ASICs and FPGAs while some are more general such as CPUs and GPUs which are normal computer hardware. CPUs are very general purpose as they are intended for every type of computing a normal computer would do without much of the optimizations that other more specialized chips can provide. GPUs, while intended for graphic processing, also have a lot of overlap with machine learning optimization as they are made for parallel processing which lends itself really well to machine learning. Some GPU producers also add additional optimizations such as tensor cores in NVIDIA GPUs. ASICs are chips that are directly made for a specific purpose. If these are made for machine learning purposes then it can greatly improve the operating efficiency. FPGAs are essentially field programmable gate arrays, essentially it allows the engineer to program it into a specific hardware configuration. These are especially useful in machine learning platforms where extra performance is needed but the system is still being worked on or needs to be updated. They are also especially useful for prototyping machine learning systems. Overall when it comes to different hardware for machine learning each has their own use case, with GPUs being the most all around useful for normal consumers as they are in most computers nowadays. FPGAs are very interesting when it comes to machine learning as even though they may not be quite as specialized as an ASIC would be, they are much more flexible and allow for further improvements to the system as the design of the system changes. This is a very unique position to be in terms of hardware as the other options are almost purely static in terms of performance while FPGAs can be continuously updated and refined. It also helps that with FPGAs some of the work on a previous chip can be translated over to a newer better chip. This gives another way to improve the performance of a system with less work as if you upgrade the FPGA itself the same program for the previous one will most likely also work, this lends itself very nicely to quick moving adaptable systems where the extra computing power is needed or wanted.

Estimated Speedup of FPGA ML

The main purpose of dedicated machine learning hardware is to make the deep learning process and subsequent classification much faster. With the added speedup it makes the process of improving the system much easier as most times the system needs to finish training before it can be refined but it can also allow for more complex machine learning systems to be implemented. Many of the largest machine learning/ AI systems such as GPT used dedicated hardware, with GPT in specific using thousands of NVIDIA GPUs. However for more application specific machine learning systems a smaller piece of hardware might be better. This is where things such as FPGAs and ASICs come into the scene. FPGAs provide a great platform for machine learning as they allow for much higher performance compared to something more general such as a CPU and also allow for constant change in the system which wouldn't be possible with an ASIC. For proof of the speedup dedicated hardware has a study was done in 2017 by Chao Huang et al.   that implemented a CNN(convolutional neural network) onto an FPGA to see the effect it had on the efficiency of the system. They tested the SqueezeNet system against multiple other works on FPGAs but also against other hardwares such as an Intel 7700k CPU and a GTX 980 Ti GPU. They found that the system implemented on an FPGAin terms of timing had an FPS of about 273.97 compared to the CPUs 19.87 than what was implemented on the CPU. This means that it could classify an image in about .0036 seconds on an FPGA compared to an image in about .05 seconds. This is a massive increase in speed over the conventional CPU and is faster by almost a magnitude of ten. Another major improvement in this system is the power efficiency in which it is about 52 times more efficient than the CPU and 4.3 times more efficient than the GPU. These findings show just how much of an improvement FPGAs can have on the speed and efficiency of machine learning. With all of these improvements how does implementing a machine learning system on an FPGA actually work? When implementing machine learning on an FPGA, there are a couple major problems that pop up, one of the biggest is with memory and the memory accesses between the different layers. In a paper written by A. Shwahana et al. They lay out some of the challenges and necessary precautions when implementing machine learning on an FPGA. First they lay out the need for FPGA accelerators as machine learning becomes larger in scale the amount of calculations can raise into the billions of operations with millions of parameters. This can and will exceed the computing capabilities for less optimized systems. The main struggle with FPGAs is the memory limitations, with such large scale systems FPGAs can't reasonably handle the amount of data that needs to be stored, this leads to the need for the excess data to be loaded in from an external memory drive. This leads to the need to balance how this is transferred and shared to ensure that FPGAs are viable for the specific system.

When looking at the speedup that ASICs have over FPGAs it is important to realize that ASICs are essentially hyper customized, highly efficient circuits that are usually made for one specific task or system. This is in stark contrast to how FPGAs are highly adaptable and reprogrammable. This difference in the core concept of the two different types of chips will obviously mean that ASICs are best for very specific machine learning tasks that would be static in nature and not needed to be continuously updated and improved upon while FPGAs would obviously be a better fit in a system where it would need to be changed like in a self driving car for example. This means in a straight head to head between the two different types ASICs will most likely come out ahead. There are a couple factors that can affect this difference in speed, ignoring the obvious answer of just using a more powerful FPGA. According to a paper written by E. Nurvitadhi, et al. The main way to close the gap in the extra efficiency provided by ASICs is to make special use of the hard blocks in the FPGA such as the DSPs. This method takes out some of the inefficiencies introduced through the reprogrammable blocks. This paper also gives a quantitative analysis of the different hardwares used for machine learning. They tested a machine learning algorithm against multiple different CPUs,GPUs, FPGAs and ASICs. They found that the ASICs they used compared to the same size FPGA with the same architecture the ASIC was about 4.5x faster due to the higher clock frequency. They also found that the ASIC was 11x more power efficient than the FPGA, this could be due to the difference in transistor size from 14nm on the ASIC to 20nm on the FPGA. The paper also stated that the increase in speed between an ASIC and an FPGA with similar specs the ASIC could be up to an order of magnitude more efficient than the FPGA. Overall, they found that with the different types of NNs they tested they found that the FPGA64 was about 5 times as fast as the baseline CPU they used and was only marginally faster than the different GPUs they tested. They also found that the ASIC64 is about 15 times faster than the baseline CPU. Which means the ASIC is about 3 times faster than a similarly sized FPGA. This example shows the major increase in performance between more general machine learning hardware and the hyper specialized with there being an almost 15 times increase in performance between a CPU and an ASIC. This means that between a CPU and an FPGA there can be an expected increase in performance of at least 300% and between an FPGA and an ASIC there could be an expected increase in performance of around an additional 300%-500%. These values are of course subject to change if more powerful hardware is used but between similarly powerful chips this could be a reasonable expectation.

How Long it Took Our Simulations to Classify an Image With Expected Speedup

During the creation of our CNN simulations we made a handmade CNN that could classify image from the MNIST database, this section of the project is entirely software based and was made in C++, to connect this aspect of the project to FPGA machine learning we will also take the FPGA specified from the FPGA specifications block to estimate the expected speedup it will have on our simulations. First the baseline for this approximation is the time it took our neural network to classify an image. The time it took our FCL model to classify a single image was .0032s. The time it took our TCL model to classify a single image was .11s. The next thing to check is the critical path through the neural network as extra branching paths will slow down the implementation in the FPGA. However in our model there is only path that follows first through the convolution layer, then the pooling layer, then finally the fully connected layer. For the TCL model there is a tensor contraction layer between pooling and fully connected layer. The amount of mathematical operations in the FCL model is 60830 additions and 60164 multiplications. In the TCL model there are 177070 additions and 205764 multiplications. The difference in the amount of calculations is the main reason for the huge disparity in classification time between the two models. The next thing to look at is the suggested FPGAs and estimate the hardware speedup. The three suggested FPGAs all don't have digital clock manager modules. The only FPGA to feature more than one core is the Altera Cyclone V 5CEA9. When it comes to clock frequencies each FPGA has a much different value, the Altera Cyclone V 5CEA9 has a clock frequency of 12.5MHz, the Altera Stratix V GSD8 has a clock frequency of 800 MHz, and the Virtex 7 VC709 has a clock frequency of 710MHz. Looking at these three FPGAs they would all be very good for implementing our system onto hardware and would all most likely increase the speed and efficiency of the system. For my estimation I will use the Alter Stratix V GSD8 as it has the highest clock frequency. The Altera Cyclone V would also be a good budget option as the second core would help with the parallelization of the convolution operations. However, I think that the drastically increased clock frequency would outweigh that benefit. Without actually implementing our simulations onto the FPGA it is impossible to know how much of a difference the implementation on an FPGA would make but it would be safe to assume a fairly large increase in performance and efficiency. I would estimate that by implementing our neural network model onto this FPGA we would see an decrease in the amount of time it would take to classify an image by about half. This would mean taking the base classification time of .0032s and turning it into .0016s. This estimation is less than the speedup that the FPGA provided in the paper by E. Nurvitadhi, et al, which was about a 5x speed increase. But with an amateur implementation such as ours, the full power of the FPGA would most likely not be accessed.

Works Cited

C. Huang, S. Ni and G. Chen, "A layer-based structured design of CNN on FPGA," 2017 IEEE 12th International Conference on ASIC (ASICON), Guiyang, China, 2017, pp. 1037-1040, doi: 10.1109/ASICON.2017.8252656. keywords: {Field programmable gate arrays;Hardware;Parallel processing;Pipelines;Memory management;Acceleration;FPGA;CNNs;Data quantization;Pipeline;Structured design;SqueezeNet},

A. Shawahna, S. M. Sait and A. El-Maleh, "FPGA-Based Accelerators of Deep Learning Networks for Learning and Classification: A Review," in IEEE Access, vol. 7, pp. 7823-7859, 2019, doi: 10.1109/ACCESS.2018.2890150.
keywords: {Deep learning;Field programmable gate arrays;Neural networks;Hardware;Acceleration;Convolution;Throughput;Adaptable architectures;convolutional neural networks (CNNs);deep learning;dynamic reconfiguration;energy-efficient architecture;field programmable gate arrays (FPGAs);hardware accelerator;machine learning;neural networks;optimization;parallel computer architecture;reconfigurable computing},

E. Nurvitadhi, D. Sheffield, Jaewoong Sim, A. Mishra, G. Venkatesh and D. Marr, "Accelerating Binarized Neural Networks: Comparison of FPGA, CPU, GPU, and ASIC," 2016 International Conference on Field-Programmable Technology (FPT), Xi'an, China, 2016, pp. 77-84, doi: 10.1109/FPT.2016.7929192. keywords: {Neurons;Random access memory;Biological neural networks;Field programmable gate arrays;Graphics processing units;Hardware;System-on-chip;Deep learning;binarized neural networks;FPGA;CPU;GPU;ASIC;data analytics;hardware accelerator},