Optimizing Neural Network Energy Consumption

Reilly Goddard

4th Year Project Report
Artificial Intelligence and Computer Science
School of Informatics
University of Edinburgh

2021

Abstract

Recent advances in GPU compute performance and efficiency have driven a rapidly expanding machine learning (ML) field. With each generation chipset bringing higher performance per Watt, many learning tasks are optimized for these new architectures. However, GPU compute needs are quickly outpacing Moore's law [27], often leading to a shortage of GPUs and rapid construction of new data centers. This lack of availability has caused many systems to continue using legacy hardware for compute needs. On average, 40% of the hardware deployed within data centers is over 3 years old [9], raising concerns over hardware efficiency and power consumption. This paper aims to provide context for understanding ML performance across various generations of hardware, benchmarking its efficiency, then extrapolating the results to discuss different hardware configurations. Currently the ML industry lacks easily accessible tools to compare systems efficiency, many opting to look only at raw performance metrics. Meaningful comparisons between separate generations of hardware and the understanding of its limitations will allow for better optimization of existing and future models. This will counteract the rapidly increasing power demands brought on the industry by cutting edge research and big data.

Table of Contents

Chapter 1

Background

1.1 GPU History

The graphics processing unit, or GPU, has existed in various forms since the 1970's. Originally it functioned as a programmable processing unit separate from the CPU, handling graphical output for a system. Early GPUs were used in various devices such as arcade cabinets, consoles and home PC's. One of the first notable GPU releases was by IBM in 1981 [33]. It was contained in the IBM PC and called the Color Graphics Adapter, supporting an output of 640x200 in 4bit color with 16kb of RAM. This was followed shortly by the Professional Graphics Controller and Enhanced Graphics adapter in 1984, marking the first GPU targeted for professional CAD and 3D work [33]. GPU development continued until 1995 where the first consumer level chips were released by companies such as Matrox, Creative, ATI, and Nvidia. The release of these GPUs along with the development of the VGA Graphics finally brought modern graphics cards to the mass market. Despite the competition from manufactures, in 2000's the market was cornered entirely by ATI (AMD) and Nvidia with their Radeon and GeForce lines of graphics cards. This brand dominance still remains 20 years later without much outside competition.

1.2 GPU Hardware

Modern graphics cards have a fairly standardized set of components, such as the GPU, video memory (VRAM), and voltage regulator module (VRM) [30]. These components are contained on a single Printed Circuit Board (PCB), commonly interfacing with CPU using the PCIE protocol. The card is affixed into a PCIE slot on the motherboard providing a high speed link between the two components.

The GPU die typically occupies the largest amount of space on the PCB, being placed directly in the middle of the board. The GPU, much like a CPU, contains billions of transistors [16], and handles large computing tasks such as rendering video or graphics outputs to a display. This draws the highest amount of power in a typical system falling between 100W – 300W [15]. The heat generated by this chip must be actively cooled

to prevent thermal throttling, a safety measure where the GPU automatically slows to reduce temperatures, inadvertently reducing performance in the process [2].

The GPU itself contains multiple smaller cores, each serving dedicated functions. Nvidia labels these as Cuda Cores, with each core falling under the category of shader processing units, texture mapping units, render output units, ray tracing cores, or Tensor Cores. They are all optimized for different functions but the massive quantity of cores allows for fast parallel computing [22].

As the GPU processes data, it needs not only a location to read incoming data from, but also a physical location to store output data such as rendered frames. The next component in the stack, the VRAM, surrounds the GPU on the PCB providing a highspeed, read and write connection directly to the GPU [30]. This memory runs at much higher speeds than typical RAM, having extremely high data thruput, allowing for the simultaneous read / write function that the GPU requires. The VRAM chip set consumes the majority of the remaining power when running the graphics card.

The VRM handles the power delivery from the power supply (PSU), dispersing it around the PCB to each component. It takes the input of 12V stepping it down to around 1.1V to 1.3V using multiple layers of MOSFET with PWM switching [30]. Aside from the power modulation, the VRM also serves to reduce noise from the input. This function is essential to the running of GPU chipsets as they have razor thin voltage margins. Deviations can result in system instability and crashes

1.3 Compute With GPUs

While CPUs are mostly applicable for problems that require parsing through or interpreting logic in code, GPUs are designed specifically for rendering a graphical output. This workflow requires highly parallel computation, dedicating more transistors to processing data compared to flow control [22]. The parallel nature also results in a high ratio between memory access and computation, dramatically reducing the need for caching. This process facilitates the low latency and high volume of computation necessary for rendering graphics in real time.

At its core, rendering uses the same base calculations of matrix multiplication and transformations, as neural networks [22]. The similar nature of these processes as well as the scalability found in learning tasks results in the GPU perfectly suiting its computational requirements.

While training a neural network, inputs are given and processed in hidden layers. Weights are then adjusted during training with the model delivering a prediction. This process is repeated multiple times with the weights and biases adjusted until model performance is deemed acceptable. Rapid calculations at the large scale required for fast model training necessitates the use of many cores so data processes in parallel instead of sequentially. The structure of the GPU perfectly fits this need, making it the preferred tool for training neural networks.

Released in 2006, Nvidia's Compute Unified Device Architecture (CUDA) was the first general purpose framework to allow parallel computation on Nvidia's GPUs [22].

It supports various languages such as C, C++, Java and Python. CUDA functions by splitting tasks that can be executed in parallel into blocks. These blocks are then passed to the GPU for computation and are able to execute on any core. The number of GPU cores varies, but they significantly outnumber the core count of CPU's resulting in a vast performance delta.

In recent years the focus of ML tasks have shifted from basic computational models to that of highly complex networks with thousands of hyperparameters. Between 2012 and 2018 the amount of computational resources used by deep learning research is estimated to have grown over 300,000x [25], leading to an equaly large increase in energy consumption. As these networks increase in depth and complexity, the resources required to optimize them also rises substantially. This has birthed two general approaches for learning, the concept of Green AI and Red AI [25].

Much of the AI industry operates at the bleeding edge of technology, searching for accuracy above all else. Red systems are purely focused on optimizing results, often spending massive amounts of compute time to improve models by fractions of a percent [26]. This consumes large quantities of energy without resulting in any significant improvement over earlier iterations of the model. There is a point of diminishing returns in optimization that is ignored in the pursuit of perfection in many cases.

Contrasting this, Green AI focuses on finding the point of diminishing returns and only optimizing hyperparameters to that point, a conscious decision to conserve energy. Green AI advocates also call for more publishing of energy consumption and efficiency figures within academic papers. Even without changing current practices, the publishing of these figures would draw attention to the issue. It is not intended to take the place of Red AI, but to supplement it when appropriate. Being mindful of energy consumption pertaining to GPUs can save system resources, allowing for more models to be run on lower spec hardware [25]. However there still remains a lack of information regarding how to specifically decrease power consumption while still maintaining acceptable levels of performance.

1.4 GPU Power Consumption

Original Equipment Manufacturer (OEM's) develop Graphics Cards, detailing the specifications for each component, the layout of the PCB, and the thermal package required to cool the GPU. Board Partners then take these specifications for each card, modifying them in an effort to improve the end result. These modifications, reduction in card size, increased cooling capacity, or increase in GPU / VRAM clock speeds, create a wide range of cards for the end user to choose from.

The power consumption of Graphics Cards, measured in Watts (W), describes the power draw that the entire GPU, VRAM, and VRM stack pulls from the wall. The manufacture of each card provides a metric called the Thermal Design Power (TDP) detailing this power draw. It serves as not only an indicator or the power draw but also the cooling solution required to run the card efficiently [2]. However, TDP functions more as a loose guideline than as a definitive metric. Depending on a system's total available power and cooling, GPU's often exceed the TDP. This happens for one of two

reasons, momentary spikes or a sustained GPU and VRAM overclock. However, both require that systems are equipped with a power supply (PSU) with adequate headroom. Not leaving a margin will result in unstable performance and constant crashing.

Despite this, TDP has remained similar across comparable Graphics Cards from multiple generations, see Table 2.1 for reference. The improvements in GPU compute have not come from simply increasing the TDP, rather they have come from advancements in GPU architecture [27]. These improvements have focused on shrinking the size of transistors, allowing for more to be packed onto each die. The smaller transistor size allows for more efficient function of the chipset, while the increased number of transistors increases compute performance [27]. These functions balance out, resulting in a more powerful GPU while keeping the TDP equal to or lower than the previous generation.

1.5 GPU Benchmarking

The change in compute performance and TDP between each generation chipset presents a problem when trying to quantify performance improvements. Benchmarking serves as a tool to compare various aspects of a GPU. These benchmarking tools work by running a standardized test, recording a metric, and computing a result [1]. Different metrics can be used when benchmarking a system. Nvidia often touts metrics like Tera Floating Point Operations Per Second (TFLOPS), Tera Operations Per Second (TOPS) in their presentations. This can then be compared with other results, allowing for comparisons between different systems or hardware configurations to be made.

Many current benchmarking suites focus on graphics performance, such as running tests using prefabricated environments within game engines or rendering a standardized scene in a CAD program [5]. These are repeatable tests that are accessible and easy to use, requiring no setup outside of the instillation of the suite. The metrics used in these tests are often very rudimentary, average frames per second while moving through a scene, or time to complete a CAD render. However, these tests are optimized specifically for graphics-based tasks, not accounting for other factors, such as power consumption or cost of hardware. Each of these factors directly affect the performance, or the efficiency of the system [27] while testing, leading to an incomplete representation of system performance.

MLPERF is the most popular benchmark for measuring a systems performance over a wide range of ML specific tasks. The benchmark, developed by industry leaders at Nvidia, Google, Stanford, and Harvard, focuses on ML training and associated tasks. These include image classification, object detection, natural language processing and reinforcement learning [24]. The tests are curated by a team of researchers from both big tech and academia, stating that they aim to accelerate progress in machine learning, encourage innovation, and keep benchmarking affordable to all [24]. It remains open source and free to use, though is not user friendly for those unfamiliar with its code base.

The benchmarks are measured using a timing method, allowing for a model to be trained, timed, and compared with other systems [24]. To account for variability in

tests, the benchmarks are run a predetermined number of times, with the score then averaged to give the final result. According to the documentations scores typically have a deviance of +/- 5% between different runs [24], larger variations indicating issues with the benchmark.

Despite its popularity it is not necessarily the only option for a ML benchmark. The lightweight AI-Benchmark is a python based tool originally developed for use with smartphones [8]. It has recently made its way to desktop benchmarking, having the advantage of a simple instillation from its pip package within python. However, due to its intended use on smartphones, the tests are far to short in duration and run at a very small scale. It does not adequately stress a powerful GPU during tests, but remains a decent choice for lower end hardware.

A better option is found when looking at TensorFLow based benchmarks. Of the few available the most popular is the ResNet-50 benchmark [29]. It is similar to a CNN test within MLPERF, however it is easier to install with the given documentation. It also remains is used as a common validation test for systems that will be utilizing TensorFlow while training models. The TensorFlow package depends on other instillation such as CUDA, so running the benchmark is a quick way to ensure the environment is ready for development. It runs on the command line and outputs a value in images per second, based on how many were processed by the GPU.

For a benchmark that is not specifically related to ML, the Blender Benchmark is an widely used option. It offers high duration testing that scales well over multiple GPUs, along with configurable parameters. It runs within Blender, an open source 3D toolkit used from professional animation of feature films to 3D modeling [3]. It utilizes a proprietary rendering ending called Cycles that features support for Nvidia CUDA. The benchmark focuses on rendering a static scene or high poly-count 3D object, such as a BMW sedan or a classroom. The benchmark tests both the GPU and CPU with a realistic load that mirrors the intended use case of somebody working within the Blender Suite.

Scores from these benchmarks are derived measuring the time that it takes to completely render the object or scene. Multiple test scenes are provided at varying levels of complexity and polycounts, allowing for the user to select a benchmark that will closely match their own work. Blender also remains free to use and completely open source, allowing anybody to run these benchmarks [3]. It also supports multiple operating systems and users can also chose to report benchmark scores to an online database that aggregates the data for all to see.

Without actually running a ML specific benchmark, Blender has the most similarities out of any other benchmarks to a ML workload. Blender heavily relies on GPU compute for rendering much like the training of a deep learning neural network. It also supports multiple GPU's in both single system configurations or in rendering nodes dispersed across a local network [5]. This also draws parallels to high end servers, used for large ML projects, with multiple GPU's or even multiple discrete systems used together. Though the benchmark on its own does not provide a meaningful score relating to ML performance, the score can be looked at broadly to estimate ML performance relative to other systems that have also run the Blender Benchmark.

Chapter 2

Procedure

This project aims to build off the work done by members of teams at MLPerf and TensorFlow, expanding the scope of data analyzed to include power consumption figures. The process for collecting data will be discussed in the following section, along with some metrics that will be analyzed. The goal is to create a metric that is indicative of both performance and efficiency.

2.1 Preparation

Modern graphics cards often contain the same GPU in a variety of different models, with minor differences such as clock speed, amount of VRAM, and the number of cores enabled. A variety of cards from Nvidia will be selected for testing. These will come from the previous 4 generatoins of GPUs, released between 2015 and 2020. From each generations the highest performing consumer card will be tested. The selected Graphics Cards can be seen within Table 2.1.

Graphics Card	GPU	Process Size	Cuda Cores	Tensor Cores	VRAM	Clock Speed	TDP	Year	Cost
RTX 3090	GA102	8nm	10496	328	24GB	1395/1695 MHz	350W	2020	£2000
RTX 2080ti	TU102	12nm	4352	544	11GB	1350/1545 MHz	250W	2018	£1000
GTX 1080ti	GP102	14nm	3584	Null	11GB	1480/1580 MHz	250W	2017	£500
GTX 980ti	GM200	28nm	2816	Null	6GB	1000/1075 MHz	250W	2015	£300

Table 2.1: The specifications and price paid for each GPU used in testing as defined in Nvidia's reference specifications [11] [12] [13] [16], with cost determined by price paid for each GPU.

Next a benchmark was selected to use as a baseline test for comparing each generation of graphics card, the ResNet-50 [29] benchmark. It runs using the TensorFlow package in Python 3 typically within a Docker container to simplify the environment setup and to promote consistency between tests. The benchmark also allows for tailoring to each GPU's capability with image batch sizes that can be specified for each run. The maximum image batch size depends on the card's VRAM capacity, with a higher capacity allowing for more images. The benchmark measures the GPU's ability to

process images within a convolutional neural network (CNN), giving a score in the form of images processed per second, or IPS.

CPU	Cores	Clock	Chipset	RAM	Clock	SSD	Power Supply
i9 7920x	12	4.4GHz	x299	8x8GB	3200MHz	8TB NVME	1600W

Table 2.2: The components which will be used in the test bench while researching.

For all testing and data collection, the same test bench will be used to keep the hardware variances at a minimum. The specifications of the test bench can be seen in Table 2.2. The system is configured in a way that it will not inhibit the performance of the GPU while testing. It offers adequate airflow for cooling the system, and components that will not inhibit the GPU's performance. The OS chosen to run test was Windows 10 due to many GPU monitoring utilities only support this Windows.

2.2 Testing

Prior to testing, the graphics card is installed in the system and checked briefly to ensure it functions. Then the card's current power limit is noted in the Table 3.1. This power limit will serve as the baseline for testing, with subsequent tests run at a reduced level. Then the environment for data collection is set up using Nvidia's version of Docker [20] within Ubuntu. This variant enables GPU acceleration, a necessity for testing. It was initialized using the following command to pull the Docker container, along with the necessary codebase from TensorFlow's Github [29] for running the benchmark.

```
nvidia-docker run --gpus all --shm-size=48g -it --rm -v
   cri nvcr.io/nvidia/tensorflow:20.12-tf2-py3
wget https://github.com/tensorflow/benchmarks/archive/
   master.zip
unzip master.zip
cd benchmarks-master/scripts/tf_cnn_benchmarks/
```

Once the environment is ready, the testing can begin, but first the maximum batch size must be determined. This is dependent on the amount of VRAM the GPU has, shown in Table 2.1, as the data set must be loaded into the memory. Once it has been found, the batch size is noted in the description of Table 3.1 pertaining to the GPU being tested. The benchmark is then run using the following code, with a batch_size of XXXX set to the value found during the preliminary testing.

```
python tf_cnn_benchmarks.py --num_gpus=1 --batch_size=
   XXXX --model=resnet50 --variable_update=
   parameter_server --data_format=NCHW --use_fp16
```

Additional software, GPUz [28] and ICUE [4], are required to run while testing. These monitoring tools will be used to track parameters outside the scope of the benchmark's result. They will first be used as a method of validation to ensure that the set power

limit is adhered to, but will also track GPU utilization, GPU clock speed, VRAM utilization, and the total system power consumption. Each of these figures serve as indications of the load put on the GPU by the benchmark. When the tests are run, it is expected that they will show between 90% - 100% VRAM and GPU utilization. A figure consistently less than this indicates that parameters for testing have not been properly optimized, skewing the results of the test. If this is the case, further modifying of the parameters may be necessary before testing.

Each test at a power limit is run three consecutive times, with a 1 minute gap in between tests to allow the temperature of the system components to stabilize. This ensures that heat does not build up, causing the GPU to thermal throttle. After the tests are complete the IPS values are added to Table 3.1. Then the power limit is reduced by 10W using Nvidia's System Management Interface (nvidia-smi), a utility included with the GPU's drivers. It allows for hardware monitoring and the modification of GPU parameters such as the power limit. The following command is used to adjust this, where XXXX represents the new power limit in watts.

```
nvidia-smi -i 1 -pl XXXX
```

This concludes the testing procedure, repeating it for each GPU until the lower bound of its power limit is reached. It is worth noting that the TDP of the card does not necessarily indicate the highest power limit. Often cards are capable of surpassing this by around 20-30W. It is also important to ensure that the benchmark runs on the correct GPU if multiple cards are present in the system.

2.3 Analysis

Once the selected cards have been tested, the data will be evaluated based on a number of factors. First, the raw performance figures of each GPU will be discussed. It is expected that as generational improvements occur, this metric will increase substantially. This data will then be considered along with the power consumption of the card. To accomplish this the cards will be compared using a Performance Per Watt (PPW) metric.

The new metric will normalize the card's performance based off of its power consumption. This will allow a for meaningful comparison between vastly different raw performance figures. It provides a quantifiable measure for comparing the card's efficiency. Then data collected from the GPU monitoring programs will be analyzed to see what effect lowering the power limit has on the GPU's components, aiming to demonstrate why performance decreases as the power limit is lowered. Finally, other aspects of performance will be analyzed, taking into account factors such as GPU cost and power cost over time while operating in a ML environment.

Chapter 3

Results

3.1 Data Review

The data collected from the testing can be seen in Table 3.1. It is split into 4 sub tables, each representing one GPU. The description of the sub table indicates the GPU used as well as the batch size used during the test. The tables record the power limit set for testing under the Watts column. Results are recorded under the columns marked Test 1, Test 2, and Test 3, with the value reported in IPS.

The column labeled Average is derived by taking the mean of the values in the Test columns. This is done to account for any variances between tests, though all tests showed less than 5% variance between runs. The PPW column also was derived based on other values in the table. It takes the Average column and divides it by the Watts column, giving a measure of Images per Watt per Second (IPW).

Each graphics card was tested over a different range of power limits, starting at the card's default power limit down to the lowest allowed by nvidia-smi. The RTX 3090 had the highest range, with the tests between 350W and 180W. The other cards operated within a lower power limit range, all starting at 270W, and dropping as far as 110W. The values between 270W and 180W were tested using all graphics cards, while other values outside this range lack data from one or more cards.

In addition to data from the benchmark, supporting data from the RTX 3090's testing has been included in Table 3.2. The Watts column represents the specified power limit, showing the same range of values as Table 3.1a. The other data in the table was collected using the GPUz utility to log parameters of the card while testing. After setting the power limit, logging was started in GPUz and run through the duration of all 3 tests. However, due to the minute-long gaps, the log file also included data from in-between tests. To combat this issue rows were dropped from the file by filtering the Memory Usage column. This left only data from the periods of time when the benchmark was actively running. Then the data was consolidated by averaging the values of each parameter at a specific power level.

Watts	Test 1	Test 2	Test 3	Average	PPW
350	1249.67	1250.86	1248.56	1249.69	3.57
340	1245.47	1242.13	1241.36	1242.98	3.65
330	1240.28	1237.64	1235.07	1237.66	3.75
320	1231.17	1230.69	1230.55	1230.80	3.84
310	1221.71	1223.60	1222.24	1222.51	3.94
300	1212.95	1211.77	1215.69	1213.47	4.04
290	1203.87	1199.51	1198.45	1200.61	4.14
280	1187.54	1195.96	1168.82	1184.10	4.22
270	1176.98	1170.95	1169.67	1172.53	4.34
260	1162.42	1157.76	1156.92	1159.03	4.45
250	1127.97	1128.94	1121.43	1126.11	4.50
240	1094.05	1092.60	1087.85	1091.50	4.54
230	1032.50	1027.25	1027.91	1029.22	4.47
220	973.79	973.07	969.40	972.08	4.41
210	893.71	885.52	881.87	887.03	4.22
200	819.47	817.18	809.33	815.32	4.07
190	738.07	730.12	726.04	731.41	3.84
180	626.51	628.26	623.34	626.03	3.47

(a) RTX 3090 (batch_size = 512)

Watts	Test 1	Test 2	Test 3	Average	PPW
270	643.70	642.69	636.14	640.84	2.37
260	634.56	640.26	632.54	635.78	2.44
250	639.16	622.30	628.28	629.91	2.51
240	636.18	621.73	624.43	627.44	2.61
230	627.37	630.67	617.45	625.16	2.71
220	627.39	631.10	606.56	621.68	2.82
210	598.86	628.75	628.39	618.66	2.94
200	612.48	609.36	614.19	612.01	3.06
190	613.37	616.69	594.21	608.09	3.20
180	584.24	616.20	605.58	602.00	3.34
170	591.54	608.80	594.31	598.21	3.51
160	591.76	587.97	593.07	590.93	3.69
150	582.58	581.30	584.44	582.77	3.88
140	541.14	538.83	535.60	538.52	3.84
130	494.49	498.38	489.11	493.99	3.79
120	425.54	430.33	424.90	426.92	3.55
110	341.46	339.72	345.91	342.36	3.11

(b) RTX 2080ti (batch_size = 128)

Watts	Test 1	Test 2	Test 3	Average	PPW
270	255.53	258.97	257.9	257.46	0.95
260	254.87	255.34	255.89	255.36	0.98
250	254.29	253.93	253.97	254.06	1.01
240	249.84	252.91	250.18	250.97	1.04
230	243.93	244.08	244.11	244.04	1.06
220	242.14	241.37	241.07	241.52	1.09
210	239.40	239.77	238.56	239.24	1.13
200	237.71	239.14	237.29	238.04	1.19
190	234.12	236.21	235.92	235.41	1.23
180	231.88	232.26	231.50	231.88	1.28
170	226.64	228.19	227.98	227.60	1.33
160	223.55	227.13	224.89	225.19	1.40
150	209.47	206.61	208.96	208.34	1.38
140	187.63	186.84	186.63	187.03	1.33
130	162.30	161.02	160.21	161.17	1.23
120	135.56	134.82	135.92	135.43	1.12

(c) GTX 1080ti (batch_size = 128)

Watts	Test 1	Test 2	Test 3	Average	PPW
270	154.98	156.84	153.21	155.01	0.57
260	153.10	154.22	151.04	152.78	0.58
250	151.45	150.70	148.43	150.19	0.60
240	150.16	147.83	148.45	148.81	0.62
230	147.45	146.69	146.19	146.77	0.63
220	146.56	145.79	145.51	145.95	0.66
210	145.83	144.47	145.19	145.16	0.69
200	145.30	143.23	143.81	144.11	0.72
190	142.34	141.87	140.97	141.72	0.74
180	135.83	136.36	136.24	136.14	0.75
170	126.76	128.53	125.42	126.90	0.74
160	115.49	113.50	113.88	114.29	0.71
150	103.36	102.40	99.46	101.74	0.67
140	90.78	87.74	86.02	88.18	0.62
130	77.35	74.58	77.89	76.60	0.58

(d) GTX 980ti (batch_size = 32)

Table 3.1: Data collected from the TensorFlow ResNet-50 benchmark

Each column in Table 3.2 shows a valuable piece of information relating to either performance or validation of the tests. The first three columns show the GPU's clock speed, VRAM speed, and temperature, all attributes that directly affect performance. The remaining columns show data pertaining to test validation. The aforementioned Memory Usage column shows a constant value of 23071MB due to the repeated tests all running at the same batch size, equal to 96% of the cards total VRAM. The GPU Load column indicates the amount of resources currently in use by the card, ranging from 95% to 98% across all tests. Both of these values show that resources were nearly fully utilized while testing. Inconsistent values in these columns would raise suspicion of inaccurate results.

The Power Draw column was obtained by measuring the actual power consumed by the card while testing. This differs from the set power limit, which acts more as a guide-

line that the GPU follows, however it is possible for the power limit to be exceeded. This occurred frequently with power consumption between +/- 20W of the power limit while under load. Though not shown in Table 3.2 due to averaging the values, the RTX 3090 peaked at 368W while testing the 350W power limit. The card never exceed the power limit for more than a few seconds at a time, often dipping below it afterwards. This is further evidenced by the average value, showing that the card would operate between 0W - 10W under the power limit for all tests. While not shown with the other cards for the sake of brevity, the GPUz statistics were monitored for constancy and followed the same trends while testing.

The decrease in power consumption also resulted in decreased performance for all cards. This was the expected outcome, as reducing the power limit will force some components to limit power consumption. When looking at the GPU MHz column in Table 3.2, a decrease in the clock speed of the card is observed at every subsequent power limit. This closely mirrors the decrease in average performance shown in Table 3.1a. As the GPU clock speed is slowed, the performance suffers because the card is computing cycles at a slower rate. This also affects the temperature, with the RTX 3090 recording a 10 degree reduction in temperature between its maximum and minimum power limits.

Watts	GPU MHz	VRAM MHz	Temperature	Memory Usage MB%	GPU Load %	Power Draw
350	1852.83	1187.7	70.62	23071	96.05	340.47
340	1836.52	1187.7	71.92	23071	96.01	331.47
330	1810.64	1187.7	70.79	23071	96.29	324.29
320	1802.32	1187.7	70.22	23071	96.21	314.66
310	1771.23	1187.7	70.25	23071	96.44	305.61
300	1764.72	1187.7	69.70	23071	96.60	297.67
290	1721.90	1187.7	69.61	23071	96.13	285.86
280	1699.14	1187.7	68.10	23071	95.76	275.91
270	1668.26	1187.7	67.86	23071	96.11	268.35
260	1613.15	1187.7	65.87	23071	95.92	257.29
250	1533.96	1187.7	66.18	23071	96.85	247.94
240	1462.53	1187.7	64.08	23071	96.67	237.82
230	1387.60	1187.7	63.91	23071	96.64	227.92
220	1285.46	1187.7	63.55	23071	96.96	216.61
210	1165.35	1187.7	62.41	23071	97.44	207.64
200	1044.36	1187.7	61.70	23071	96.74	197.90
190	902.79	1187.7	60.53	23071	97.38	188.11
180	793.82	1187.7	60.48	23071	98.27	179.90

Table 3.2: The RTX 3090's average stats and power consumption figures recorded with GPUz while running the benchmarks

Despite the GPU clock reduction, VRAM clock speed remained constant for all tests with the RTX 3090, indicating that the card prioritizes memory speed and bandwidth over GPU cycles when conserving power. This was unexpected as a major factor in the card's 350W TDP was the increase to 24GB of VRAM. When designing the card Nvidia was unable to procure 2GB GDDR6 chip sets, instead using 24 individual 1GB modules. While this was the standard for past generations of GPU, cards such as the RTX 2080ti only required 11 of these chips. The additional 13 modules used on the

RTX 3090 dramatically increased the power consumption, peaking at 95W. This stayed consistent for all tests, with VRAM accounting for 50% of the power consumption at the 180W power limit. A similar observation of a static VRAM clock speed was observed when testing the other GPUs.

3.2 GPU Performance

Within Figure 3.1, two distinct groupings can be observed, the RTX 3090 / 2080ti and the GTX 1080ti / 980ti. There is a large performance deficit between the two groups, with the RTX cards exhibiting far higher performance compared to the older GTX cards. This occurred for a multitude of reasons, primarily the addition of Nvidia's Tensor Cores to the RTX cards. These new cores are specifically designed to accelerate matrix operations, a key aspect of ML's computation. This is compounded by generational improvements made by reducing the transistor size in the manufacturing process.

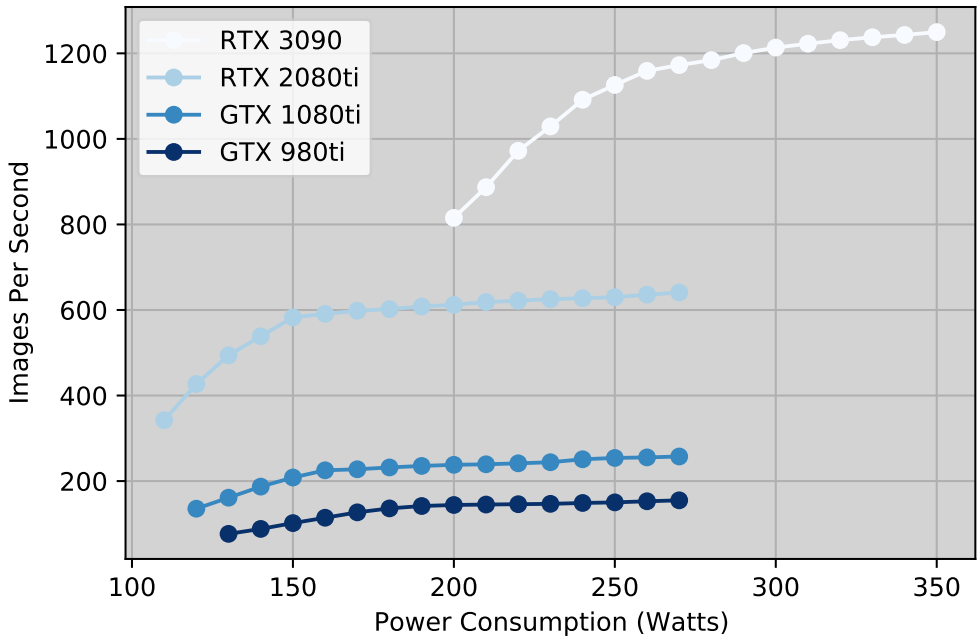

Figure 3.1: A comparison between the raw performance of each graphics card within the Tensorflow ResNet50 CNN test.

The highest performance observed from any card was during the 350W test preformed on the RTX 3090, averaging to 1249.69 IPS. This performance is nearly double that of the RTX 2080ti's highest score of 640.84 IPS. When looking at the individual specifications of the two cards, as shown in Table 2.1, the RTX 3090 utilizes 10496 Cuda Cores and 24GB of VRAM, a 141% and 118% increase respectfully over the RTX 2080ti. These increased specifications account for some of the performance gain observed. However following the same logic would suggest that power consumption

should increase proportionally, more than 100W. Additionally the RTX 2080ti actually has 216 more Tensor Cores than RTX 3090. This raises suspicion that another factor must also be responsible for the increase in performance.

In this case, differences in the GPU architecture are responsible. Both GPU's in question come from Nvidia's new RTX line of cards, though there were significant changes made between the 2080ti's Turing and the 3090's Ampere architectures. The manufacturing process used for each card can be seen in Table 2.1. Amprere uses a new 8mm process from TSMC, while Turing used an older TSMC 12nm process, just a slight modification to the previous 14nm process. As discussed in Section 1.4, the decreasing in transistor size increases the efficiency and density of the chip, allowing for the increase in Cuda Cores while only causing a 1.3% increase in the GPU's physical chip size [15].

The Tensor Cores introduced to the consumer market with Turing are actually the second generation [14], as they were first used in Nvidia's professional only Volta GPU's. When Nvidia released the specifications and performance figures for Amprere's third generation of Tensor Cores, they stated that each core had 2x the performance per clock of the second generation [31]. This would explain the roughly 40% decrease in tensor core count, while increasing the overall performance.

The same trends observed within the RTX series cards are also present for the GTX series cards. The large gap in performance when testing can be attributed to lack of tensor cores, lower total cuda core count, and older manufacturing processes. The GTX 980ti, GTX 1080ti, and RTX 2080ti, all possess the same TDP, shown in Table 2.1. Despite this, the GTX cards perform far worse than the RTX cards. The GTX 980ti, the oldest GPU tested, showed the lowest IPS and IPW metrics. It scored only 155.1 IPS at a power limit of 270W.

During the testing of the GTX 980ti, it appeared that the thermal limitations of the card were reached when the power limit was set above 200W. This may be due to the age of the card or the cooling characteristics specific GPU being tested, but the other GPU's did not encounter the same issue. Even the RTX 3090 was able to successfully manage thermals at its 350W power limit. However the discussion of raw performance figures fails to account for other factors such as the cards power consumption relevant to performance.

3.3 Performance Per Watt

The doubling in performance observed with the RTX 3090 does come with a cost. More specifically it comes a 100W increase to the TDP of the card. Nvidia understands this when releasing new GPU's, often touting figures such as a 1.9x increase in FPS/Watt in games [15]. This metric falls under the category of a Performance Per Watt (PPW) metric, something that becomes increasingly important when quantifying differences between system performance at large scales.

The data in Figure 3.2 was generated by plotting the Watts column against the PPW column from Table 3.1. This PPW metric, called IPW, shows the relative performance

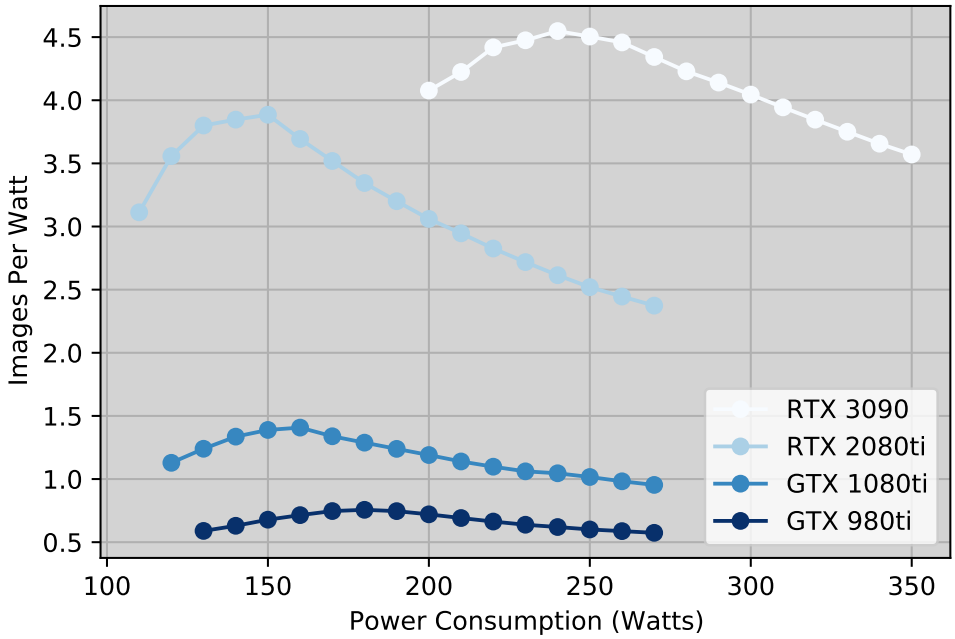

Figure 3.2: The normalized performance of each GPU from Figure 3.1. It was calculated by dividing the images per second by the power limit.

of the cards compared to one another with respect to power consumption. When comparing data in both Figure 3.1 and Figure 3.2, it is clear that as power consumption increases, diminishing returns in performance are observed. In the case of the RTX 3090, peak performance occurs at 350W with 3.57 IPW, while peak efficiency occurs at 240W with 4.54 IPW.

Each GPU has a different power limit corresponding to peak efficiency. This was expected as variations in the architecture and GPU specifications would result in a mixed outcome. However, all GPUs while testing showed peak efficiency at a power limit between 60% and 70% of its TDP.

As the power limit for each card is initially decreased, the IPW metric increases at an apparently linear rate. This continues until it peaks, then quickly falling off at lower power limits. When looking at the data in Table 3.1, each GPU shows a lower IPW metric at its maximum power limit compared to its lowest. This curve creates an interesting optimization problem when planning systems for ML applications. It would be unexpected if Nvidia was unaware of their GPUs efficiency curve, indicating that they consciously pursue extra performance at the expense of efficiency.

Chapter 4

Discussion

4.1 Optimizing System Performance

Depending on the use case, optimal system configurations may vary dramatically, with factors such as system budget, physical size, existing infrastructure, and thermal or power limitations all needing consideration. The wide variety of possible system combinations means that it is impossible to offer a 'One size fits all' solution when planning. Instead, each of these factors should be investigated individually and prioritized. Many system configurations may offer similar levels of performance to one another but at the expense of one or more categories. Often when selecting components the user must make compromises to fit the system within the given constraints. The Min-Max strategy commonly tackles this problem.

A min-maxed system is one that focuses exclusively on 1-2 aspects of the system, maximizing their values while disregarding or minimizing others. This often results in a system that may be unbalanced or severely lacking in some categories. An example of a min-maxed system would be one that focuses purely on raw performance, while minimizing physical space taken. Something like Nvidia's recently announced DGX A100 [17] system falls under this category. It contains 8 x Ampere GA100 GPU's connected through a 600GB/s NVLINK connection, each GPU containing over double the raw compute performance of a RTX 3090. All of this fits into a single 6U server chassis, roughly 2x the size of the Corsair Obsidian 500D used in the test bench for this paper. Though it is unparalleled in compute power density, it is the result of proprietary engineering done by Nvidia and does not come cheaply, costing roughly £212,000.

Even a more balanced system approach for a large scale compute server may involve some compromises. If financial constraints are a high priority, building a system which involves less proprietary hardware and even used equipment, may better balance the end result. Many data centers and consumers often sell older hardware at far below the original MSRP. This creates opportunity for those looking to save financially, however these products have often been used for years ahead of time and will have no manufacture warranty. As discussed in Section 2.2, it is important to look at the power consumption of GPU's compared to performance, especially as the models get older. While it may be possible to currently purchase 10 GTX 980ti cards for the same price

as a single RTX 3090 and achieve similar performance metrics, the power consumption would increase by a factor of nearly 10 as well. In addition to the increased power consumption, the physical size of a system speced in this way would be substantial, likely requiring multiple 4u server racks. The decision of using a single GPU vs a multi GPU system warrants further discussion.

4.2 Single GPU Systems

Single GPU systems offer a myriad of advantages over multi GPU systems. Primarily they are more streamlined, requiring less physical space, smaller power supplies, and lower CPU overhead compared to a multi GPU setup. This results in a system that is essentially 'Plug and Play', as most GPU intensive tasks will run very efficiently on a single card setup. Software designed to run on GPU's often requires specific implementation to run on multi-GPU systems, also increasing the overhead and troubleshooting for the developer. Applications without this implementation may not take advantage of a second GPU, leaving it to idly consume power in the system. This actually occurred when testing with the TensorFlow benchmark, despite two of the same card being present in the system, the benchmark crashed when run or would only utilize the GPU in a single card. Many hours were spent trying to troubleshoot this issue and an optimal solution was never reached, hence the presence of only single card performance metrics for this benchmark.

When releasing a consumer GPU, such as the cards used in this paper, Nvidia assumes that most users will only utilize a single card in their system. This is evidenced by the slow removal of SLI or NVLINK multi GPU capabilities from Nvidia's consumer products. While it used to be a widely available feature, only the RTX 3090 supports it from Nvidia's current generation of GPU's. Therefore they are operating under the assumption of a single card use case for the majority of users, attempting to maximize performance in this configuration. When looking back at the data shown in Figure 3.1 through this lens, it begins to make more sense. The performance gains from increasing power consumption after the card reaches maximum efficiency are mild, however they add up to a significant value once the TDP is reached.

The RTX 3090 achieved peak efficiency around 240W, shown in Figure 3.2, with a raw performance of 1091.5 IPS. Comparing this to its maximum performance of 1249.6 IPS at 350W gives a difference of 158 IPS. This performance increase of 14% over peak efficiency comes at the cost of a 45% increase in GPU power consumption. However, being a single GPU system, the user may opt for the performance increase despite the lowering of efficiency. When looking at system performance overall, the GPU only increases total power consumption by 110W. Comparing this to Nvidia's recommended power supply configuration of 750W - 1000W [16], depending on system's other components, the previously large 45% increase now only accounts for a 14% - 11% power increase system wide. This value now closely matches the performance gain observed in the card, essentially keeping the system's PPW value the same, despite the decrease in GPU efficiency.

Due to these factors, in smaller systems dedicated to ML research, it is advised that a

single GPU configuration is used, however this does come with an exception. GPU's while used for compute in this context of this paper are often required to perform other functions in a system such as display output. During preliminary tests with the RTX 3090, it was found that performance dropped by nearly 10% on average when the GPU being tested was also tasked with handing the display overhead. This used a portion of the GPU's compute capability as well as roughly 1GB of VRAM. The simplest way around this is to use a low powered secondary GPU for the primary purpose of handling display outputs, though this is only relevant if the system requires a GUI. A system that utilizes a remote monitoring protocol such as SSH avoids this overhead entirely.

4.3 Multi GPU Systems

Despite the drawbacks mentioned in the previous section, there are still merits to multi GPU systems. The largest benefit is often the increased memory capacity. As mentioned in the previous sections, Nvidia utilizes a multi GPU protocol called NVLINK, allowing for the sharing of many GPU resources, including VRAM pooling. Despite the absence from most new consumer cards, NVLINK is still widely used in data center applications and supported by Nvidia's professional products. The technology allows for two or more GPU's to communicate over a high bandwidth physical link between the cards. The connected GPU's are then able to act as a single unit, pooling both the compute and memory capabilities.

The amount of VRAM present in a system often dictates the ability of a GPU to handle large workloads efficiently. As discussed in Section 2.2, the amount of VRAM each card had available dictated the batch sizes used in testing. Further increasing the batch size resulted in the GPU exceeding memory capacity and the test failing. Ideally the data set used when training a model would be less than the total VRAM available. However, unlike the system memory, VRAM is soldered to the GPU's PCB, making it nearly impossible to increase. Nvidia is aware of this issue, offering GPU's such as the RTX A6000, a professional GPU based on the same GA102 chip as the RTX 3090, but with 48GB of VRAM compared to 24GB. These cards come at a significant price increase over their mainstream counterparts, but do fix the memory issue. In extreme cases this may not be enough still requiring more than one GPU to utilize memory pooling for large data sets. If implemented correctly, NVLINK works well in most situations, though it is currently only supported for two cards in the case of the RTX A6000 and RTX 3090. If more compute is still required, additional GPU's can be used in a different manner.

Multi GPU systems that lack a hardware bridge between cards instead work in a parallel computing manner with the system's CPU managing the overhead. Unlike an NVLINK setup, the GPUs are separate without resource pooling, allowing them to execute tasks in parallel independent from one another. This scales to larger than 2 GPU's, being mainly limited by the CPU's PCIE lane capacity. The additional overhead also requires more system memory and explicit multi GPU implementation for the model training. Despite these drawbacks, the scale can be increased dramatically through distributed computing. In this use case, systems can each contain a number of

GPU's that are then connected to one another over high speed local networking. This allows for each subsystem to be self contained, having its own processors, RAM, and GPU's, while still contributing performance to a larger task. While scaling across these systems is never perfect, recent techniques to mitigate performance loss by roughly 30% [7] have been proving successful.

When looking at the power consumption of these multi GPU systems, the GPUs take up far more of the system's total power consumption compared to the single GPU system. This in turn increases the importance of GPU efficiency for data centers. Assume a data center had a distributed computing network containing 10 nodes, each with 8 RTX 3090 GPU's tuned for maximum efficiency, with an additional 750W used by each node for the system overhead. They could expect around 10 * ((8 * 240W) + 750W) = 26.7kW in total power draw from the servers. If they then decided to run the cards at the recommended 350W TDP, the servers would draw an additional 8.8kW, a 32.9% increase in power consumption. Compared to the expected performance gain of 14%, this power increase would lower the total system's efficiency. In addition, one of the primary costs in running a data center is electrical. Then by increasing power consumption by 32.9%, it is expected that running expenses would increase in a similar amount.

Lastly, multi GPU scaling will never reach true linear rates as increases in system complexity and the use of distributed computing will all lower the final performance figures. This is due to the lack of bandwidth between systems and GPUs. Within a computer the GPU has the largest bandwidth to the processor it is directly connected with. Other GPUs in the same system are limited in connection speed by needing to utilize the connection with the processor as a bridge between the two. This lowers the bandwidth each GPU will receive and increases latency. This is one of the main reasons NVLINK is used as it bypasses this CPU managed connection in favor of a separate, high bandwidth hardware path.

4.4 Example System Configurations

Below in Table 4.1, multiple system configurations are detailed, focusing on balancing not only performance characteristics, but also energy consumption and cost. The systems, while different each have their own distinct benefits over one another. It also introduces a variation on the IPW metric to help understand energy costs over a longer period of time. This is the Images per Watt Hour (IPWH) metric, obtained by multiplying IPW * 3600 seconds. The IMG / Hour column shows the total number of images processed in an hour assuming the system is constantly running. For the sake of simplicity it assumes perfect scaling across multiple GPU's.

Systems A - G are sorted in the table in accordance to the IMG / Hour metric. This order also matches the initial system cost, with it scaling almost linearly to performance. Historically this has not always been the case, however recent increases in demand from sectors such as Bitcoin mining have skewed the pricing structure. As each GPU is known to produce a certain amount of Bitcoin per day, GPU prices reflect this ability, fluctuating with the currency. Despite this, looking at the initial cost of the system plus

System	GPU	Num	Cost	Watts	IPWH	IMG / Hour
A	RTX 3080	4	£8,000	1400	12,852	18,000,000
B	RTX 3080	4	£8,000	960	16,344	15,696,000
C	RTX 3080	3	£6,000	1050	12,852	13,500,000
D	GTX 980ti	16	£4,800	2880	2700	10,368,000
E	RTX 2080ti	4	£4,000	1080	8,532	9,216,000
F	RTX 2080ti	4	£4,000	600	13,968	8,380,800
G	GTX 1080ti	8	£4,000	1280	5,040	6,480,000

Table 4.1: Example system configurations to illustrate the differences between optimizing performance, efficiency, and cost.

the power consumption over time yields interesting results. The median price per kWh in the United Kingdom falls at 15p [23], or 0.015p per Wh of energy consumed. This allows pricing models to be constructed over a period of time using the same system configurations.

System	1 Month	6 Months	1 Year	2 Years	4 years	8 Years
A	£8,153	£8,919	£9,839	£11,679	£15,358	£22,716
B	£8,105	£8,630	£9,261	£10,522	£13,045	£18,091
C	£6,114	£6,689	£7,379	£8,759	£11,518	£17,037
D	£5,115	£6,692	£8,584	£12,368	£19,937	£35,074
E	£4,118	£4,709	£5,419	£6,838	£9,676	£15,352
F	£4,065	£4,394	£4,788	£5,576	£7,153	£10,307
G	£4,140	£4,840	£5,681	£7,363	£10,727	£17,455

Table 4.2: The cost over time of each system as defined in Table 4.1.

Shown in Table 4.2 is the cost of running the GPUs over various periods of time. One notable configuration is F, coming in at the lowest cost after 8 Years, this is due to its low power consumption of 150W per GPU. It was set at this level to optimize efficiency as found in Section 3.1. Configuration D is also worth mentioning as it utilizes 16 GPU's. The initial cost of this system was low due to the age of the GTX 980ti, but upon running the system it requires 2880W, the most of any configuration. The total cost after 8 years was also the highest of any configuration.

This data can be further expand on to create a performance to price metric for the systems. This is done by taking the total images processed over a period of time and dividing that by the cost of running the system to up to that point. This will illustrate which systems offer the best value from a financial perspective. In Table 4.3, it is System E that takes the lead when only run for a single month.

Though it may be tempting to run the GPU at its TDP, as the short term cost is mostly determined by the purchase of the card, in the long term it will end up costing more.

System	1 Month	6 Months	1 Year	2 Years	4 years	8 Years
A	1.610	8.835	16.018	26.991	41.050	55.506
B	1.413	7.962	14.840	26.123	42.142	60.778
C	1.610	8.835	16.018	26.991	41.050	55.506
D	1.109	5.089	7.935	11.014	13.666	15.536
E	1.6333	8.569	14.895	23.608	33.367	42.060
F	1.504	8.353	15.332	26.329	41.051	56.982
G	1.137	5.836	9.946	15.348	21.071	25.900

Table 4.3: The number images (Millions) processed per £ spent over time.

Looking at systems F and B, both start out at a lower value compared to their less efficient counterparts. However, after 1-2 Years have past, they surpass the other systems and continue to widen their gap as time progresses. Systems A and C show the same values because they are essentially the same configuration with C simply removing a GPU. Due to the nature of this metric, the number of cards in the system is irrelevant for the calculations as it is canceled out when dividing by the cost.

4.5 Variations in Graphics Cards

Each of the GPUs used in testing, while developed originally by Nvidia, were actually built off of a provided specification by board partners. As discussed in Section 1.4, these partners are responsible for tweaking the provided GPU designs in various ways. The core specifications such as number of Cuda Cores and amount of VRAM remains constant across all models, but other aspects such as power delivery components, cooling designs and even the TDP of the card are modified frequently.

While theses design choices may not appear significant at first, they can greatly impact the functionality of the card. In multi GPU systems where cards are often placed directly next to one another blower style coolers or water cooling blocks may provide a better thermal management solution than the standard fans found on most cards. Without taking into account the thermal characteristics of the specific card being used, the system is at risk of thermal throttling due to heat buildup, ultimately resulting in lower performance than expected [2].

The chassis that will contain the GPUs is also extremely important to consider. Case size and designs vary widely between manufactures, often dramatically affecting performance. A case that has inadequate airflow will always preform poorly due to the inability to exhaust heat from the system again leading to thermal throttling [2]. The physical size of the case may also limit GPU selection due to the form factor of the card. This was an issue encountered when sourcing GPUs for this paper, with some models being too long to fit in the case.

Chapter 5

Evaluation

5.1 Project Scope

The project as initially proposed over a year ago actually remained very similar to the final outcome of the paper. However the completion was not without issue and constantly changing plans. The primary problems stemmed from the testing of GPUs, as multiple failed attempts to collect data occurred over the span of a month. Originally testing was to be conducted with MLPERF but many issues were encountered when attempting to setup the testing environment.

Immediately after acquiring the 8 GPUs, the test bench had issues with a corrupted SSD as the systems boot drive. Windows was then reinstalled and updated to the most recent developer build on a new SSD. In reality this was a good thing as the instillation was void of any unnessicary programs, but the process still took nearly a week between troubleshooting OS issues, ordering the replacement hardware, and reinstalling the OS.

After the hardware was stable, multiple attempts were made to install various benchmarks from MLPERF [24]. However when installing the required Docker container for a benchmark, the download speed on the initial pull was below 500Kbps giving an estimated 22 day wait time despite a download speed of 300Mbps under normal circumstances. Multiple solutions were attempted such as purchasing a Docker subscription, but nothing increased the speed. It is suspected that the ISP was throttling traffic when pulling the container. Attempts were made to install the required dependencies for a test directly, but this too failed.

Moving on from MLPERF after over two weeks of troubleshooting, the next best option appeared to be the TensorFlow ResNet-50 benchmark [29]. Once the dependencies were installed, surprisingly the docker container hosted by Nvidia [19] downloaded smoothly. Once installed the procedure developed in Section 2.2 was followed. However, unexpectedly the GPU failed to pass through to the Docker container. After another full day of troubleshooting it appeared that the recently applied update from the Windows Developer program WIP OS 21327 broke GPU support in WSL2 (Windows Subsystem for Linux) [21]. This was indicated in Nvidia's WSL2 Cuda Toolkit Documentation [18], again halting progress until the next update fixed the issue.

At this point the testing was already nearing a month behind schedule, but a working environment was finally set up. The initial plan involved testing not only the single card configurations of each GPU from Table 3.1, but also dual card configurations. This would have provided a secondary data point when referencing performance and is in line with many systems used for ML research.

This data would have paired well the discussion in Sections 4.3 and 4.4 as they both involved discussion on the benefits of using multi GPU systems, but lacked actual first party data to back up the claims. Instead, multiple papers [32] [7] [10] were read, taking their findings into consideration when discussing the systems.

Whenever the benchmarks were run with more than a single GPU physically in the system, the program would throw massive amounts of python errors, indicating troubles with GPU compatibility were still present. In addition, when starting the docker containers, the functionality enabling the selection of which GPU should be passed into the container was broken. In total over a month was spent simply troubleshooting what should have been a simple testing setup for collecting data.

5.2 Continuation of the Project

Further research in the field of GPU power efficiency should be conducted. As stated in the paper on Green AI vs Red AI [25], power consumption figures are hardly present in academic works on ML research. A publicly available database containing information on GPU power consumption figures would help many researchers further optimize the performance of their hardware. This would require the collecting data from papers which already show PPW figures, as well as encouraging other researchers to participate with future projects.

Another aspect this report lacks is an explanation as to why the GPUs peak at a specific power limit. This would involve further research, delving into the electrical engendering aspect of GPU design. Electrical Engineering is not the intended background for a reader of the paper, catering towards a more traditional computer science reader. Still, why the GPU hardware behaves in a specific way when exposed to varying power limits would provide further evidence backing up the results discussed in Section 3.1. The effect that transistor width has on GPU core count and thermal limitations should also be explored further. Unfortunately due to the aforementioned time constraints this research was cut, prioritizing more central topics and data collection.

5.3 Conclusion

This paper attempts to educate the reader on a series of complex topics involving GPU power consumption and performance optimization. It covers the information while trying to display it in a clear and concise format. The performance figures shown in Table 3.1 were collected over a series of benchmarks run using TensorFlow. The data covers four separate GPUs, each representing a different generation of graphics card from Nvidia, ranging in release dates from 2015 - 2020.

The RTX 3090 was the most most modern GPU tested and subject of further analysis. Performance data collected with the GPUz [28] utility while benchmarking helped to shed light on the inner working of a GPU at different power levels. This data, shown in Table 3.2, displays a decrease in GPU clock speed similar to the performance deficit observed in the benchmark. However, the VRAM clock remained constant through all testing, suggesting that the GPU prioritizes high memory speeds over that of the core clock. The temperature also dropped significantly between the 350W and 180W tests.

In situations where a GPU may be prone to thermal throttling this indicates that reducing the power limit would help to contain the heat generated by the card [2]. This would affect performance, but it can be mitigated by first finding the GPU's optimal power limit. This level can be seen for each GPU in the 3.1 under the PPW column in each subtable. The value was obtained by dividing the Average column by the Watts column, giving a performance per watt metric.

Using a PPW metric is an essential component to building and testing modern systems used for artificial intelligence research. It helps to find tune the power consumption of the system, while still maximizing performance. It was shown in Section 4.4 how the different power limits and hardware configurations affect not only power consumption but also long term costs of running the system. Unfortunately many academic studies still fail to record information pertaining to power consumption, instead focusing on performance and optimization of networks [25].

Within the six years between 2012 and 2018, the amount of computational resources dedicated to ML applications grew in excess of 300.000x [25]. This increase is expected to continue growing exponentially. In a recently published report, it is estimated that data centers account for 1% of global energy consumption [6], growing to 30% by the year 2030. This increase does not mean data centers will be using energy saved by other sectors, rather it indicates that total global energy consumption will rise in proportion.

The ability to properly optimize these data centers, ensuring maximum efficiency, will be an essential skill for all researchers in the industry. Failing to optimize a data center could result in a further 20% - 30% increases in power consumption as indicated in Section 3.3. It is imperative that researchers are aware of their power consumption going forward, making a conscious effort to keep it at a reasonable level.

Advances in both hardware and methodologies for training neural nets has kick started a new era of machine learning. Models are showing previously unparalleled levels of accuracy when training on the ever generating supply of data. However, these improvements in accuracy have become dependent on the availability of massive computational resources. These massive cloud computing data centers continue to expand, consuming more energy with every new server. The new models are costly to train both computationally and environmentally. If steps are not taken to ensure these data centers offset emission by ensuring data centers receive energy from sustainable sources, their carbon footprint will grow dramatically in the coming years. A new mindset is needed within the industry, one of compromise that will balance performance figures while attempting to minimize environmental impact. A new era of Green AI is need to ensure a sustainable, technologically advanced civilization for generations to come.

Bibliography

[1] David Abdurachmanov, Peter Elmer, Giulio Eulisse, and Robert Knight. Future computing platforms for science in a power constrained era. *CoRR*, abs/1510.03676, 2015.

[2] Ganapati Bhat, Suat Gumussoy, and Ümit Y. Ogras. Power-temperature stability and safety analysis for multiprocessor systems. *CoRR*, abs/1806.06180, 2018.

[3] Blender.org. Blender, a free and open soruce 3d creation suite. `https://github.com/tensorflow/benchmarks/tree/master/scripts/tf_cnn_benchmarks`, 202.

[4] Corsiar. Icue unite your setup. `https://www.corsair.com/uk/en/icue`, 2021.

[5] Maximilian Denninger, Martin Sundermeyer, Dominik Winkelbauer, Youssef Zidan, Dmitry Olefir, Mohamad Elbadrawy, Ahsan Lodhi, and Harinandan Katam. Blenderproc. *CoRR*, abs/1911.01911, 2019.

[6] Hayley Dunning. Cloud computing could be producing hidden greenhouse gas emissions. `https://www.imperial.ac.uk/news/199433/cloud-computing-could-producing-hidden-greenhouse/`, 2020.

[7] Nilanjan Goswami, Amer Qouneh, Chao Li, and Tao Li. An empirical-cum-statistical approach to power-performance characterization of concurrent gpu kernels. 2020.

[8] Andrey Ignatov, Radu Timofte, Andrei Kulik, Seungsoo Yang, Ke Wang, Felix Baum, Max Wu, Lirong Xu, and Luc Van Gool. AI benchmark: All about deep learning on smartphones in 2019. *CoRR*, abs/1910.06663, 2019.

[9] Peter Judge. The data center life story. data centers are born, they get old, and some day they die. `https://www.datacenterdynamics.com/en/analysis/the-data-center-life-story/`, 2017.

[10] Deepak Narayanan, Mohammad Shoeybi, Jared Casper, Patrick LeGresley, Mostofa Patwary, Vijay Korthikanti, Dmitri Vainbrand, Prethvi Kashinkunti, Julie Bernauer, Bryan Catanzaro, Amar Phanishayee, and Matei Zaharia. Efficient large-scale language model training on gpu clusters. 2021.

[11] Nvidia. Geforce gtx 900 series. `https://www.nvidia.com/en-gb/geforce/900-series/`, 2015.

[12] Nvidia. Geforce gtx 1080ti ultimate geforce. https://www.nvidia.com/en-sg/geforce/products/10series/geforce-gtx-1080-ti/, 2017.

[13] Nvidia. Geforce rtx 2080ti rtx. it's on. https://www.nvidia.com/en-gb/geforce/graphics-cards/rtx-2080-ti/, 2018.

[14] Nvidia. Nvidia turing gpu architecture graphics reinvented. https://www.nvidia.com/content/dam/en-zz/Solutions/design-visualization/technologies/turing-architecture/NVIDIA-Turing-Architecture-Whitepaper.pdf, 2018.

[15] Nvidia. Geforce rtx 30 series graphics cards: The ultimate play. https://www.nvidia.com/en-us/geforce/news/introducing-rtx-30-series-graphics-cards/, 2020.

[16] Nvidia. Geforce rtx 3090 the bgfgpu. https://www.nvidia.com/en-gb/geforce/graphics-cards/30-series/rtx-3090/, 2020.

[17] Nvidia. The world's first ai system built on nvidia a100. https://www.nvidia.com/en-gb/data-center/dgx-a100/, 2020.

[18] Nvidia. Cuda on wsl user guide. https://docs.nvidia.com/cuda/wsl-user-guide/index.html, 2021.

[19] Nvidia. Ngc containers. https://ngc.nvidia.com/catalog/containers/nvidia:tensorflow, 2021.

[20] Nvidia. Nvidia docker, nvidia container toolkit. https://github.com/NVIDIA/nvidia-docker, 2021.

[21] Nvidia. Warning please do not update to wip os 21327 if you would like to use wsl with gpu support. https://forums.developer.nvidia.com/t/warning-please-do-not-update-to-wip-os-21327-if-you-would-like-to-use-wsl-with-gp170087, 2021.

[22] Bogdan Oancea, Tudorel Andrei, and Raluca Mariana Dragoescu. GPGPU computing. *CoRR*, abs/1408.6923, 2014.

[23] UK Power. What is the average electricity unit rate per kwh? https://www.ukpower.co.uk/home_energy/tariffs-per-unit-kwh, 2021.

[24] Vijay Janapa Reddi, Christine Cheng, David Kanter, Peter Mattson, Guenther Schmuelling, Carole-Jean Wu, Brian Anderson, Maximilien Breughe, Mark Charlebois, William Chou, Ramesh Chukka, Cody Coleman, Sam Davis, Pan Deng, Greg Diamos, Jared Duke, Dave Fick, J. Scott Gardner, Itay Hubara, Sachin Idgunji, Thomas B. Jablin, Jeff Jiao, Tom St. John, Pankaj Kanwar, David Lee, Jeffery Liao, Anton Lokhmotov, Francisco Massa, Peng Meng, Paulius Micikevicius, Colin Osborne, Gennady Pekhimenko, Arun Tejusve Raghunath Rajan, Dilip Sequeira, Ashish Sirasao, Fei Sun, Hanlin Tang, Michael Thomson, Frank Wei, Ephrem Wu, Lingjie Xu, Koichi Yamada, Bing Yu, George Yuan, Aaron Zhong, Peizhao Zhang, and Yuchen Zhou. Mlperf inference benchmark. *CoRR*, abs/1911.02549, 2019.

[25] Roy Schwartz, Jesse Dodge, Noah A. Smith, and Oren Etzioni. Green AI. *CoRR*, abs/1907.10597, 2019.

[26] Emma Strubell, Ananya Ganesh, and Andrew McCallum. Energy and policy considerations for deep learning in NLP. *CoRR*, abs/1906.02243, 2019.

[27] Yifan Sun, Nicolas Bohm Agostini, Shi Dong, and David R. Kaeli. Summarizing CPU and GPU design trends with product data. *CoRR*, abs/1911.11313, 2019.

[28] TechPowerUp. Gpu-z graphics card gpu information utility. `https://www.techpowerup.com/gpuz/`, 2021.

[29] TensorFlow. Tensorflow benchmarks. `https://github.com/tensorflow/benchmarks/tree/master/scripts/tf_cnn_benchmarks`, 2020.

[30] Jeff Tyson and Tracy Wilson. How graphics cards work. `https://computer.howstuffworks.com/graphics-card.htm`, 2021.

[31] Jarred Walton. Nvidia rtx 30-series ampere architecture deep dive: Everything we know. `https://www.tomshardware.com/features/nvidia-ampere-architecture-deep-dive`, 2020.

[32] Wanjing Wei, Yangzihao Wang, Pin Gao, Shijie Sun, and Donghai Yu. A distributed multi-gpu system for large-scale node embedding at tencent. 2020.

[33] XoticPC. History of graphics processing units. `HistoryofGraphicsProcessingUnits`, 2019.