Characterizing and Exploring the Reduction of Carbon Emissions by GPU Computing

Sarah Boyce
University of Virginia
abp6cm@virginia.edu

ABSTRACT

This paper attempts to determine the efficacy of integrating XGBoost [6], a powerful machine learning model, within the Chase framework to optimize GPU power consumption for training deep neural networks (DNNs) while minimizing carbon emissions. Chase, which enhances the existing Zeus framework by incorporating carbon intensity data, typically utilizes Support Vector Regression (SVR) for predicting carbon intensity fluctuations. This study compares the performance of XGBoost against SVR in this context to determine which model more effective power and carbon management. The result determine whether this change is necessary in addition to determining other helpful pieces of Chase that contribute to better carbon emission rates. The code and scripts used in this project can be found here: https://github.com/abp6cm/Zeus-Chase.

1 INTRODUCTION

The growing computational demands of training deep neural networks (DNNs) have significant environmental implications as their energy consumption continues to increase. The Chase framework addresses these concerns by adjusting GPU power settings, such as power limits and batch sizes, based on carbon intensity data, thus aiming to reduce the carbon footprint of DNN training. Traditionally employing Support Vector Regression (SVR) for carbon intensity predictions, and ruling out other prediction models thorugh experimentation, this study introduces the use of XGBoost, a gradient boosting model known for its efficiency and accuracy in various machine learning tasks, to evaluate whether it can enhance Chase's performance.

2 BACKGROUND

The evolution of deep learning has been intertwined with increased computational demands, especially with the adoption of GPUs for training deep neural networks. While delivering great advancements in AI capabilities, this progression has significantly elevated the energy consumption and associated carbon emission of such technologies. Traditional methods of mitigating these environmental impacts have included shifting computational tasks to places or times of lower carbon intensity. The methods, however, is often proved impractical because of regulatory, privacy, or logistical constraints.

Recently, innovations such as Zeus, have attempted to address these concerns by dynamically adjusting the operational parameters of a GPU, more specifically the batch size and power limit. These adjustments are done based on real time energy consumption metrics. Zeus, however, lacks in the aspect of considering carbon intensity. By ignoring this, suboptimal environmental outcomes can occur despite the improvement in energy efficiency. Therefore, Zeus is solving part of the problem, but not the problem entirely.

To close this gap, Chasing Low-Carbon Electricity for Practical and Sustainable DNN Training (Chase) was developed [8]. Chase builds on the framework provided by Zeus but integrates real-time carbon intensity monitoring directly. By leveraging short term carbon intensity forecasts, Chase dynamically adjusts GPU power limits. Carbon intensity is the measurement of carbon dioxide emissions per unit of electricity generated. It is usually measured in grams per kilowatt-hour. This definition is crucial for evaluating the environmental impact of various energy sources because it helps quantify exactly how much carbon dioxide is being released as a result of generating electricity. Fossil fuels such as coal, oil, and natural gases are all carbon intense whereas renewable sources like solar and wind have very low carbon intensities.

By using historical carbon intensity data, Chase is able to reduce carbon emissions without compromising on the training performance of a GPU. This benefits the sustainability of AI practices as well as offers a practical solution to the GPU carbon emissions issue without needing to shift workloads to other physical locations or delaying training tasks. The integration of Chase is a large step forward in reducing the carbon footprint of DNN training as well as other AI research and deployment. Because it addresses both the issues of energy consumption and carbon emissions, Chase contributes to a responsible AI development landscape. In turn this helps align technological advancement with environmental sustainability.

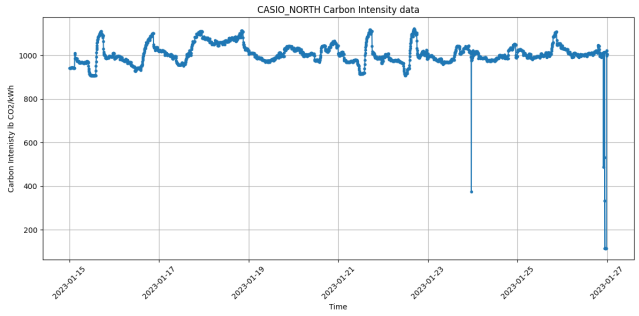

Figure 1: Plotted historical Casio North carbon intensity data

3 IMPLEMENTATION

3.1 Chase vs Zeus

While Zeus and Chase both aim to optimize energy consumption of training tasks, their approaches to the problem are different due to a difference in priorities and mechanisms. This difference is important in understanding why Chase may be a better solution to addressing specifically the carbon emissions issue.

Zeus's primary focus is on balancing energy consumption with training performance. It dynamically adjusts the power limit and batch sizes of a GPU in order to optimize the training process for energy efficiency. Zeus uses a feedback system that adjusts these settings based on observed power usage and computational throughput. Although this is an effect approach to reducing energy consumption, it does not directly address carbon emissions themselves. A reduction in energy consumption indirectly contributes to a reduction in carbon if the source of the energy comes from a consistently carbon intense source. Zeus does not take into consideration the fact that carbon intensity fluctuates during the day, as well as day by day, so it lacks the specificity to directly solve the carbon crisis. These fluctuations can be seen in Figure 1

Chase extends Zeus by incorporating carbon intensity data into the decision making process. This new method not only consider the immediate power computation like before, but now this consumption directly translates into carbon emission based on the current carbon intensity of the electricity supply to a data center. Chase uses a forecasting model to predict short term changes in carbon intensity which allows it to adjust the power limit of a GPU. This approach to managing carbon efficiency is particularly crucial as it aligns with the timing of computationally intensive tasks with moments of lower carbon intensity, such as when renewable energy is more readily available.

3.2 Effect on the Power Grid

Chase's integration of this carbon intensity provides a more targeted approach to reducing the carbon footprint of DNN training. This is especially important to regions where energy sources vary throughout time and are unpredictable. Its ability to respond to real time shifts in carbon intensity means that Chase can adjust to unexpected changes in an energy grid. Such flexibility allows data centers and research facilities to become active participants in demand response systems, which support grid stability and enhance the overall sustainability of the energy ecosystem.

Demand response systems are programs that are designed to manage the energy consumption of a grid. Instead of changing the supply of energy, it changes the demand for it. These systems are important during times of high energy usage or when the grid is under some sort of stress (like during extreme weather). Participants (or consumers) can reduce or shift their power usage during peak times by responding to signals from energy providers or grid operators. This can help stabilize the power grid and prevent outages from occurring. Data centers consume a significant amount of energy not only because of their servers but also cooling systems and other operational systems (like lights, bathrooms, etc.). Chase can significantly make a change in demand response strategies. Because Chase lowers its energy use during peak times, data centers can effectively reduce the stress on the grid, which can prevent voltage sags, outages, and the need for emergency measures. There is also a possible financial benefit. If data centers properly manage their consumption in response to grid needs, many regions will actually offer economic incentives. This ability to react real time to changes in grid conditions, or energy prices, can make a big difference in the operational costs of DNN training and its environmental impact. [5] [7]

3.3 Regression Models

The regression model employed by Chase is used to forecast carbon intensity. Carbon intensity fluctuates based on many factors (such as time of day, type of energy source, and overall power grid demand), and by predicting these upcoming changes, Chase can dynamically adjust the power limit of a GPU during training.

Chase uses these models by first collecting historical carbon intensity data for a specified location. This data can possibly include past measurements across different times and under varying conditions. (Because this information is quite difficult to get, both Chase and these experiments use data specifically from WattTime Casio (California Independent System Operator) North region as it is a free data set. Then regression models like SVR and XGBoost are trained using this historical data. The goal of this is for these models

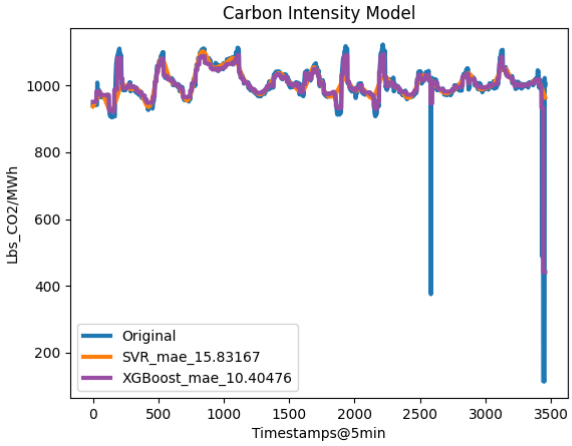

Figure 2: Two regression models predicting the future carbon intensity data of Casio North. The SVR model was previously implemented in Chase (and is the default setting) whereas XGBoost has not been seen before.

to learn patterns or trends in the changes of carbon intensity based on a smaller data set than needed. Once these models have trained, they can then predict future carbon intensity values. For example, if the model knows the carbon intensity pattern on a typical day or week, it can then forecast future values for upcoming days. Finally Chase then uses these forecasts to decide how and when to change the power usage of a GPU. If the forecast calls for low carbon intensity, Chase will schedule high energy consuming tasks during these periods or go on to perform urgent tasks. On the other hand, during high carbon intensity periods, it can throttle back GPU power usage or delay non-urgent tasks.

Many different models were tested for Chase, specifically SVR and other Boost models. An interesting model Chase does not test is XGBoost [8]. Figure 2 shows these two models with their respective mean absolute errors. The choice between these models depends on their accuracy and the specific characteristics of the carbon intensity data. SVR is effective on datasets with a clear margin of separation whereas XGBoost is known for handling large datasets efficiently. XGBoost has a lower MAE value compared to the SVR model, meaning it could be better at capturing the nuances of carbon intensity data, especially the Casio North dataset. The benefit of this could translate into more accurate forecasts which in turn could enable more effective power and carbon management by Chase.

3.4 Framework

For the sake of this paper, the Zeus framework [9] will be classified into three different versions: the first being the original release, the second being the framework downloaded and experimented on, and the third being the most up to date version.

The first Zeus framework was configured such that all important functions and operations essentially spawn from the “dataloader.py" file. All adjustments to the power limits and batch sizes occur here. Chase is forked from this version of Zeus. It follows the same dataloader structure, with a few added helper files such as “carbon.py" and other changes to the dataloader file itself in order to incorporate the carbon intensity values into its calculations.

The second version of Zeus is completely reconfigured. The dataloader file was deprecated as the functions were split across multiple files. Supposedly this makes the framework easier to edit and understand. The power monitor stayed the same. Notably, a power limit file was created in which all power limit calculations occurred as well as the actual changes in the power limit computer settings. A batch size document was nonexistent. When the Docker environment was downloaded, the second version of Zeus was implemented. Transitioning between these two to attempt to compare their differences was difficult as there were hardly any similarities. Using Zeus and Chase was almost like using two completely different frameworks.

As of a few weeks ago, Zeus has now updated their framework to version three in which there is now a file that calculates batch size and the dataloader file has been completely deleted from their repository. Because of this, it is possible the experiments done are not an accurate representation of Zeus’s capabilities as batch sizes were not considered in changing the power limit. This, however, does not affect the outcomes of Chase because it is using version one in which the batch sizes were still calculated. Due to these issues, Chase is a better framework not only because of its benefits to the carbon crisis, but also because it is operating with a batch size optimizer.

4 INFRASTRUCTURE

The experiments for btoh Zeus and Chase were conducted on a server with the specifications detailed in Table 1.

The experiments were carried out within a Docker environment to ensure consistency and reproducibility of the results. The PyTorch framework was utilized for implementing the deep learning models and conducting the training processes.

The GPU resources utilized for the experiments consisted of four NVIDIA GeForce RTX 2080 Ti GPUs. The NVIDIA System Management Interface (nvidia-smi) was employed

Table 1: Experimental Setup Specifications

Component	Specification
Server	gpusrv18
CPU	Intel(R) Xeon(R) Silver 4210 CPU @ 2.20GHz
GPU	4x NVIDIA GeForce RTX 2080 Ti
OS	Ubuntu 22.04.3 LTS
nvidia-smi	550.54.14 [NVML supported]
PyTorch	2.0.1+cu118
Docker	26.1.0

for monitoring and managing GPU power and frequency during the experiments.

5 TASKS AND CHALLENGES

5.1 Tasks

Historical carbon intensity data was collected firstly from Watttime using their API. [2] Because there are restrictions to other datasets (such as price), the collected data is from the Casio North region. [4] The data collected included date, time and carbon intensity values. This was then put into a plot along with SVR and XGBoost prediction models 2.

Using the Imagenet dataset, Zeus was successfully run on all four provided GPUs and then just one GPU (information on these can be seen in the class project submission). From there, Chase was also trained on this dataset on all four GPUs and just one using the SVR model. Then, the code was modified in the carbon.py file in order to change the model from SVR to XGBoost, which are two sklearn libraries [1]. This is a novel addition as Chase has not attempted to implement the XGBoost model thus far in training. From there, just one GPU was trained using the XGBoost model. The power consumption data from all tasks were saved to a local device and then plotted using Python.

5.2 Challenges

During the initial training of Chase on the Imagenet dataset [3], the memory of the Docker environment was exceeded and therefore crashed, stopping any and all processes from running as well as any saved data there may have been on the Docker. Thankfully enough information was saved in order to proceed with the class project, but all the results from Chase were lost. In addition, the Zeus framework is continually changing so when trying to understand the Chase framework, there was no base comparison to make to Zeus which had been in use for a month or so and was the basis of knowledge.

SVR on:	1 GPU	4 GPUs
Energy (kW)	8972665	31061988
Time (seconds)	61458	61651
Cost (gCO2)	9205238825	50055434

Table 2: These figures are each rounded to the nearest ones place. The full data can be found on the GitHub repository

6 EVALUATION

6.1 SVR: GPUs vs GPU

Training on one GPU verse four causes an extreme difference in energy consumption, specifically 9 million to 31 million respectively3. This increase, is not proportional to the number of GPUs, suggesting less than linear scaling efficiency in terms of energy. The training time is fairly consistent between the two tasks, meaning here the scaling up of the GPUs does not increase the training time as it does when running the Zeus framework. What is most notable is the drastic change in carbon cost. There is a decrease in cost from 9.2 billion when training on 1 GPU compared to only 50 million when training on four. This shows the importance of parallel processing in reducing carbon emissions (more benefits of parallel processing are given in the project report).

6.2 SVR vs XGBoost on One GPU

In the proceeding, “the SVR model" or “the XGBoost model" are referring to the training of the Imagenet data set using the Chase framework where the carbon intensity prediction models differ depending on these two options. Therefore, this comparison deviates only in the prediction model, whereas everything else (expect time of day) is kept the same.

As shown in Figure 3, at 100W, the SVR and XGBoost models while implemented in Chase are calculating its optimal power limit. Here there are a lot of spikes in data for the SVR implementation, whereas the XGBoost model has low variability. In addition, the XGBoost model takes longer to identify the proper power limit. The SVR model then decides on an optimal power limit of 155W, which is lower than the choice by the XGBoost model of 180W. During the first 500 seconds, the XGBoost model has a hard time stabilizing after finding its optimal power limit when the SVR model has less spikes in its data. This may suggest the SVR model is quicker at finding a stable and efficient operating point.

Both models show similar patterns of power usage, with peaks and valleys at regular intervals and almost aligning with each other. Because XGBoost is training with 180W of power, it completes before the SVR model, which is expected 4. It is hard to interpret which model is performing better simply by looking at these plotted graphs. Thankfully, Chase

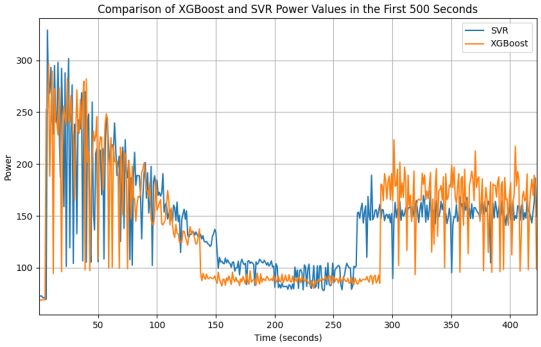

Figure 3: First 500 seconds of monitored power values during training when SVR and XGBoost are used at the prediction models in the Chase code. Power is measured in terms of watts and time in seconds. These models were not trained at the same time, simply starting together on the graph to visualize and compare their beginnings

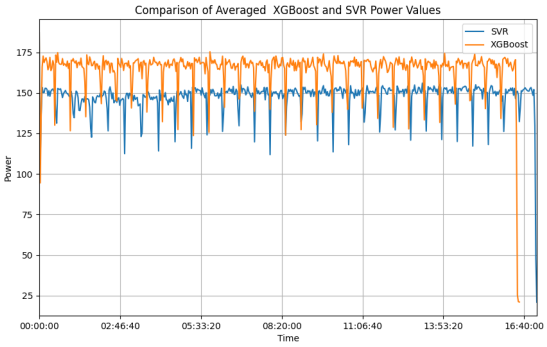

Figure 4: All collected data from SVR and XGBoost as the prediciton models. The power values are averaged over every 120 seconds (2 minutes). The drastic changes in the beginning and end of the plot show when Chase is done (or has not started) training

has a built in log function (which is another difference from Zeus) that saves useful information.

The SVR model does consume less total energy compared to XGBoost as seen in Table 3. Although there is a loss of time of 2311 seconds (or 38.5 minutes), there is an even greater gain in energy consumption (693,921 joules). If this were run on Zeus, the experiment would stop there and SVR would be considered the greener option, but Chase is interested in another metric, carbon emissions. The overall carbon cost for XGBoost is lower than that of the SVR model by 29,817,566 $gCO_2$2. So, although it consumes more energy, the XGBoost

Model	SVR	XGBoost
Energy (kW)	8,972,665	9,666,586
Time (seconds)	61,458	59,147
Cost (gCO2)	9,205,238,825	9,175,421,259

Table 3: These figures are each rounded to the nearest ones place. The full data can be found on the GitHub repository

training model trains faster and emits less carbon during training. Therefore, as predicted by our calculations 2, the XGBoost is a more accurate model in predicting carbon intensity which can translate here to even lower carbon costs. It is difficult to make a definitive conclusion here however because we did not train both models at the same power limit, so results could be different if they decided on the same one. While running Chase, there were differences in the power limit selection, even when run ten minutes apart, which is why these results may not be conclusive. Running more experiments and comparing those via averages or even comparing them running at the same time of day (as this may affect the carbon intensity values in some way) to see if the XGBoost model truly is a better option than the SVR model. These preliminary results are a great place to start experimenting more.

7 CONCLUSION

The investigation into the use of XGBoost within the Chase framework illustrates significant potential for improving the accuracy of carbon intensity predictions. These results indicate there is a future to look into with integrating different prediction model into the Chase framework and how this can truly change the carbon emissions issue at hand. Working forward in this area can help grow sustainability measures for data centers everywhere in a practical, realistic manner.

8 FUTURE WORK

As previously mentioned, future work would include updating the Chase framework to fit with the current Zeus framework. This would greatly improve the user experience when working with Chase for the first time as there is a lot more documentation on how to use Zeus. Also, it is currently difficult to understand the differences between the two frameworks by simply looking at the code, so making them seamlessly integrated with each other can be beneficial.

An avenue that would be interesting to explore is renewable energy and how to directly integrate this into the Zeus/Chase framework. As of now the choice of using renewable resources is up to the grid in which the data center derives its energy from, so being able to find an alternative to this and directly using a known renewable source would be

extremely helpful in diminishing carbon emissions by data centers.

This is an interesting topic as it needs to pull experts from many different areas outside of computer architecture or, more broadly, computer science, like environmentalists, power plant operators, electricians, engineers, etc. Collaborating with these fields can be a future endeavor for this project as computer science evaluations do not provide the entire picture.

ACKNOWLEDGEMENTS

This document is produced with the help of Yang Yang. The project largely followed the study of these works: Zeus [9], Chase [8] and related research.

REFERENCES

[1] [n. d.]. scikit-learn: Machine Learning in Python. https://scikit-learn.org/stable/

[2] [n. d.]. WattTime. https://watttime.org/

[3] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. 2009. ImageNet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*. 248–255. https://doi.org/10.1109/CVPR.2009.5206848 ISSN: 1063-6919.

[4] Josh Mahan. 2023. Understanding Data Center Energy Consumption. https://cc-techgroup.com/data-center-energy-consumption/#:~:text=in%20the%20world.-,How%20Much%20Power%20Does%20a%20Data%20Center%20Use%20Per%20Square,as%20high%20as%20300%20watts

[5] James McBride and Anshu Siripurapu. 2022. How Does the U.S. Power Grid Work? https://www.cfr.org/backgrounder/how-does-us-power-grid-work#:~:text=Introduction,from%20industry%20to%20household%20appliances.

[6] Nvidia. [n. d.]. XGBoost- What Is It and Why Does It Matter? https://www.nvidia.com/en-us/glossary/xgboost/

[7] Office of Electricity. [n. d.]. Demand Response. https://www.energy.gov/oe/demand-response

[8] Zhenning Yang, Luoxi Meng, Jae-Won Chung, and Mosharaf Chowdhury. 2023. Chasing Low-Carbon Electricity for Practical and Sustainable DNN Training. *arXiv preprint arXiv:2303.02508* (2023).

[9] Jie You, Jae-Won Chung, and Mosharaf Chowdhury. 2022. Zeus: Understanding and Optimizing GPU Energy Consumption of DNN Training. https://doi.org/10.48550/arXiv.2208.06102 arXiv:2208.06102 [cs].