The Environmental Impacts of Large Language Models

Mishika Govil
Stanford University
459 Lagunita Drive
mishgov@stanford.edu

Abstract

As computationally-intensive Large Language Models rapidly advance in their development and global usage, it is of increasing importance to evaluate their environmental effects on the world. In particular, their high energy demand is only projected to grow, establishing an exigence for exploring how LLMs might impact the environment going forward and what steps can be taken to mitigate deleterious effects caused by extreme energy usage. Over the course of this paper, we explore the litany of factors, architectural and algorithmic, that go into the determination of the energy consumption of LLMs and describe a toolkit for reasoning about environmental impacts of LLMs going forward.

1. Introduction

1.1. Guiding Questions

Over the past few years, Large Language Models (LLMs) have skyrocketed in both technical advancement and popularity. With billions of weights and parameters, these models are incredibly complex. However, enabling such complexity often requires the use of large amounts of computational power. Building on a long academic tradition of looking at the environmental impacts of novel technologies, this paper will explore the current state of research evaluating the environmental impacts of Large Language Models. In particular, the focus will be on reasons and methods for evaluating the training, and per-inference cost of different reasoning models, guided by insights gleaned from research evaluating the environmental cost of training said models. One question we seek to explore is whether different models perform with different levels of environmental efficiency when dealing with different prompts. Another question we seek to answer more broadly is how to go about measuring tangible environmental impacts in an industry where technology, design, and adoption move at incredibly quick speeds. Essentially, how do we maintain consistency in evaluating these impacts when today's LLMs may be miles ahead of yesterday's, and tomorrow's may evolve into a completely new beast to tangle with?

This paper will specifically evaluate the methodologies used in current literature to determine the energy usage and environmental impact of LLMs, focusing on a study from UMass Amherst evaluating emissions in the training stage and HuggingFace's Chat UI - Energy tool for evaluating energy consumption in the inference stage.

Given the expectation of constant innovation and change within this domain, this paper seeks to provide a toolkit with which researchers can systematically and dynamically evaluate the environmental impacts associated with different LLMs. In particular, this toolkit will identify factors and variables associated with LLM development that have direct impacts on the environment, situate those factors in the context of how they may be poised to change in the future and how sensitive the net environmental impact will be on each of those factors, and broadly identify aspects of development to account for when conducting environmental analyses of LLMs. Additionally, this paper seeks to evaluate the current methodologies being used by researchers and industry in estimating their own energy usage and environmental impact, with the goal of identifying assumptions they make and potential improvements.

1.2. Assumptions and Clarifications

For clarity, "inference cost" refers to the environmental impacts related to usage of the models via prompting. Additionally, while it is incredibly important to account for Scope 2 and 3 (indirect) emissions if possible, this paper will only focus on environmental impacts relating to the training and usage of the models themselves. Other factors that affect overall environmental impact, such as workplace infrastructure emissions, hardware manufacturing emissions, or transportation emissions, are addressed in other literature and will not be addressed in this paper.

1.3. Roadmap

This paper is broadly divided into seven key sections: introduction, background, problem, training, testing, technical deep dive, and toolkit formation, with this roadmap concluding the introduction section.

For the background section, we will begin by providing both the relevant background and the exigence for looking at the environmental impacts of LLMs through analyzing the environmental impacts of prior digital infrastructure. First, we will explore the environmental effects of the Internet and digital infrastructure before the era of LLMs, particularly looking at email and blockchain technology. Analyzing email's emissions will allow us to see how small transactions multiplied by large populations and high frequencies can have large impacts, as well as expose us to the rationale behind our first question on differentiating models and prompts. Looking at blockchain technology will shine a light on just how computationally intensive digital technologies are becoming in recent years and provide us with a comparison point of a problem with a strong literature of environmental impacts behind it. Then, we will briefly provide context on why it is both of academic and of social importance to care about the environmental impacts of LLMs.

For the problem section, we will provide an overview of the different components that go into evaluating the environmental impacts of LLMs. We will provide context for the difficulties associated with accurately estimating power usage, emissions, and associated environmental impacts. Additionally, we will introduce different language model architectures and investigate their environmental effects.

Then, we will move onto the training section, in which we look at energy consumption and emissions caused by training LLMs. We will begin with a broad discussion of the factors that go into determining the consumption and emission metrics, let that lead us into a broad analysis of the literature on LLM training and environmental impacts, and then focus on analyzing the methodology of a particular case study.

We will then conduct a similar analysis within the testing section, in which we look at the per-query energy consumption and emission values. After starting out with a discussion of how these values are calculated, we will look at the literature of test-time environmental effects of LLMs and then do our own analysis of one particular tool designed to measure energy consumption live, augmenting it with the appropriately calculated estimates for carbon emissions and water usage.

Within the technical deep dive, we will take a closer look at certain aspects of particular LLM architectures and reasoning mechanisms. The purpose of this is twofold: to understand the directions in which new innovations may take us in terms of energy and environmental impact and to understand the mechanisms through which the complexities of calculating the environmental impacts of LLMs are established.

Finally, with the insights we've gleaned from the prior sections, we will proceed to build a toolkit for current and future environmental evaluations of LLMs. Rather than one particular equation, which is inherently architecture and design-dependent, we feel a broader toolkit of considerations is a more useful and sustainable approach for these types of evaluations in rapidly-evolving industries.

2. Background

2.1. Overview

As new technologies increasingly shape our world, they also leave an impact on it. In this paper, given that we are focusing on LLMs, we will summarize previous environmental analysis of innovations within the realm of information technology, the Internet, and computer systems. As of March 2020, before the mass proliferation of generative AI tools such as LLMs, slightly less than 4% of global greenhouse gas emissions could be directly attributed to the world's technological and digital infrastructure, a number similar to the emissions attributed to the global airline industry [7]. Specifically, each internet user would have on average 912 pounds of carbon dioxide emissions attributed to them in a year [7]. While this number was not negligible, it explained why digital technology was not necessarily the most impactful sector to look at in terms of environmental consciousness. For the US, below 2% of national electricity usage went towards supporting data centers in 2020 while that number was closer to 4.4% in 2023 and is projected to grow exponentially in the coming years, with estimates going up to 12% by 2028 [7] [15]. Much of this exponential growth has coincided both with the increased technological proliferation, and thus, demands in the Global South but also with the advent of widespread, computationally intensive generative AI, particularly LLMs.

2.2. Case Study: Email

However, before discussing LLMs, it is important to understand the context in which LLM's environmental impacts should even be studied and why it makes sense to consider segmenting impacts by model and reasoning type. This idea that different Internet-based transaction types lead to different energy usage levels and environmental impacts is not a new one, which we will see through a dive into the emissions associated with a popular, seemingly mundane technology: email. Even in 2010, carbon emissions associated with just one mechanism of information transfer, email, could be further dissected into categories of emails such as spam, which released an average carbon dioxide equivalent (CO2e) of 0.3g per email, regular emails, which released an average CO2e of 4g per email, and emails with attachments, which released an average CO2e of 50g per email [7]. Rather than estimating all email emissions as some weighted average of the numbers above, an estimate that may not accurately characterize that many individual emails, this breakdown allows us to see that the majority of emissions come from a particular email type: those with attachments.

Within this paper, we will go on to explore whether a similar breakdown can be made for LLMs. Given that certain tasks, such as generating unique images from scratch, take longer and are presumably more computationally expensive than simpler tasks, such as listing the capital of California, which we will go on to discuss in the test section, we believe that it could be useful to segment energy and emissions impacts by prompt-type rather than viewing them as broadly generalizable to prompting in general. While individually many of these emission counts may seem minute, being a tiny fraction of a gram at times, multiplying them by the high frequency at which a large portion of the country or even the world sends emails leads to a much larger impact. As of 2020, with the advent of larger phones and a stronger technological culture, estimates of daily email-attributed emissions for a single person reached 3.5lb [7]. On top of that, the average business user could be estimated as contributing nearly 300lb of carbon dioxide equivalent emissions from emails annually, which, to put in context, would be akin to a typical car being driven for 200 miles [7].

While emissions from transportation are more directly and visibly understood, carbon emissions for digital applications such as LLMs are abstracted away, distanced from the average user who cannot see any direct pollution being created with their actions of prompting as they would have been able to see the fumes coming out of the exhaust pipe of their car. To truly understand the impact of just email on carbon emissions, one study by the energy company Ovo estimated that every British adult sending one fewer "thank you" email a year would save the same amount of carbon, 16.5 tons, as removing over 3000 diesel cars [7]. Just looking at the case of email gives us a great deal of insight into both how online transactions have real-world environmental implications and why the impacts should be subdivided by transaction type at the highest levels of specificity to gain the most thorough understanding of which actions lead to the most serious consequences.

2.3. Non-email Digital Infrastructure

Other digital infrastructure also contributes to energy usage and environmental damage in varying ways, including other social and messaging platforms. While SMS texts only accounted for .014 CO2e per text, digital messaging platforms had similar emissions profiles to emails, with those profiles being further complicated by the need for increased subdivision to account for different emission levels for different message types: plain text, emojis, gifs, images, and attachments [7]. While messaging generally just involves the transmission of plain text and has accordingly small per-message emissions levels, streaming videos online makes up 1% of global emissions by generating 300m tons of carbon dioxide annually [7]. In a Google report analyzing usage in 2018, it was reported that an average Google service user produces less than 8g of CO2 a day through their Google activities [7]. However, a more fitting parallel for LLM queries in terms of usage style would be internet searching, which had a reported per-search footprint of 0.2g CO2e around 2010 [7]. When multiplied by the number of searches per person and number of people making searches, this relatively small number is able to have exponentially large impacts. These examples serve to provide context on how even mundane, seemingly inexpensive tasks can add up to notable emissions when applied to a large global population.

2.4. Cryptocurrency

However, what we have seen in the past decade is that some of the rising "hottest" technologies are exponentially more computationally intensive than the simpler transactions of the early Internet. One key example is that of cryptocurrency, or more specifically, the blockchain technology powering it. As of November 2018, Bitcoin uses just under 46TWh of electricity annually, translating to roughly 22 Mt CO2 emissions per year, a number that compares to Sri Lanka's, Jordan's, and Kansas City's annual emissions [17]. The bulk of Bitcoin's emissions come from its "Proof of Work" mechanism, in which validation is based on essentially solving computationally intense mathematical problems, and within 2018 itself, the average compute power needed to solve one of these puzzles quadrupled [17]. Given these increasing numbers, a large conversation has arisen around cryptocurrency and its social, economic, and environmental impacts[2]. We are currently in the midst of the development of a similar corpus of literature for LLMs and their socioeconomic and environmental impacts, making it useful to draw on parallels from another recent computationally expensive technology.

2.5. Ethical LLM Development and Social Responsibility

As LLM proliferation grows alongside increasingly advanced model development, which requires a great deal of training, it would be an understatement to say that computationally intensive tasks will have an impact on the energy use in the world. Aside from the deleterious effects of increased energy usage and carbon dioxide emissions into the atmosphere, which we will be assuming prior context of, there are reasons to specifically care about the role LLMs will play in environmental stewardship. Research suggests that underprivileged communities may be doubly impacted by Large Language Models due to their decreased likelihood of benefiting from the progress enabled by LLMs yet increased likelihood of suffering repercussions due to the increased resource consumption needed to power LLMs [3].

When looking at certain groups' reduced propensities to benefit from LLMs, research points to language fragmentation as one particular vehicle through which this disparity in benefits is formed. To understand language fragmentation, we focus on one particular common component in LLM architectures: the tokenizer. This tool is utilized to transform the text-based input to tokens which the model is able to more easily relate to one another probabilistically. However, it has been found that non-English languages, particularly those more common in lower-income countries, require more tokens to be generated for the texts [3].

Additionally, environmental impacts are not just limited to electricity usage and carbon emissions. One key impact is water usage, as the giant data centers supporting the computations powering LLMs require a great deal of cooling, primarily done through water that needs to be acquired and transported in incredibly large quantities [16]. While at face value it may seem as though advances in LLM efficiency would come with less reason to worry about LLM-associated energy usage, it may actually be the opposite. One seemingly-counterintuitive possibility is that in fact increasing the efficiency of LLMs may contribute to higher energy consumption. This theory is motivated by the same underlying explanations for why many seemingly logical interventions seem to backfire, such as when adding additional highway lanes or bypasses actually increases traffic rather than decreasing traffic as intended. This strange phenomenon is often referred to as the Jevons Paradox, and it purports that the same advancements that increase the efficiency of certain technologies may actually encourage exponentially higher usage rates of those very technologies, so much so that overall resource consumption actually increases by sheer volume rather than decreasing by efficiency [14] [21] [20] .

Within the realm of LLMs, the release of the highly efficient DeepSeek model has raised a number of questions about what the future of model development will signal for the future of data center development. DeepSeek's reduced energy consumption entails lesser reliance on larger, centralized data centers,of the capacity of multiple hundreds of megawatts, implying lower overall electricity demand and the enabling of smaller-scale or edge data centers, around the order of magnitude of tens of megawatts [20]. While earlier models led to increased belief in the need for larger data centers, the advancement of highly efficient LLMs may contribute to a slowing of the growth of large data centers and the rapid increase of smaller or edge data centers as well as more local computing designs [20]. In fact, some researchers envision the future of model energy consumption to be a sort of hybrid format, in which larger, centralized data centers support the training of LLMs and possibly help support the most complex of inference tasks while smaller data centers, edge nodes, or local compute designs take over for typical inference tasks [20].

Given the incredibly rapid integration of LLM technologies into the business and everyday lives of millions of people, it is imperative that we obtain a strong understanding of not only the current emissions associated with current tools, but, keeping in mind the rapid rate of development in this field, develop a system of regularly keeping a pulse on the emissions associated with different, increasingly complex digital technologies.

3. Problem

3.1. Discussion of Components

When looking at the energy consumption numbers associated with large language models (LLMs), multiple avenues of accounting are needed, including energy associated with obtaining materials for and manufacturing the associated hardware, computational power needed to train the model, and computational power needed on an operational level - when the model is used [6]. For the sake of this paper, we will be diving into the last two categories - looking at the energy consumption associated with training and using LLMs. While this paper will also address the inference portion of LLM serving, it is important to note that the training portion constitutes a significantportion of the emissions from LLMs, with a neural architecture search training for a transformer clocking in at 284t of CO2 and a BERT base model using a level of energy comparable to a cross-country flight in the US [3]. As of 2024, a single query + response in ChatGPT would be responsible for over 4 grams of CO2eq, a 2000% increase from the emissions of a web query [6]. The intensive energy usage in LLMs can be understood through looking at the unique needs of LLM applications, which require intensive usage of both memory and compute.

3.2. Compute versus Memory Bound Optimization

When LLM's address queries, they do so in the two phases of prefill, in which every input token is parallel processed in order to generate the first token and context is stored in a cache, and decoding, in which the cache and token are used to generate every next token. Generally, the prefill phase makes efficient use of GPUs whereas the decoding phase has lower rates of GPU utilization [6]. While the prefill phase is compute-bound, the decoding phase is memory-bound [6]. Memory-bounded code is when the memory bus, the computer subsystem consisting of a set of conductors facilitating information transfer between the CPU and RAM, is saturated. Essentially, it is when the measured memory system performance reaches its achievable peak. On the other hand, compute bounded code is when the maximum compute instruction throughput is reached, meaning the performance of an operation is near the maximum limit of the functional unit that services that type [?]. Generally, that is used to describe programs that primarily consist of doing calculations, such as multiplying small matrices, where the CPU is primarily being used. These relatively simple repeated matrix calculations, actually enable modern AI workloads to only have compute energy comprise a tenth of their energy usage, a non-negligible but non-dominating amount [?]. Accordingly, the usage of compute energy predominantly falls within the relatively less energy intensive prefill phase whereas the memory-bounded decoding phase likely consumes a much larger amount of energy. The relevance of this is shown through the different optimization approaches that can be taken when dealing with compute bound versus memory bound programs: in the former, a common strategy involves either shifting the compute load to other functional unit types or optimizing the use of that functional type, while in the latter, focus is placed on optimizing the use of the memory hierarchy and its various component memory subsystems such as specialized cacheing including the L2 cache, read-only cache, surface memory, and shared memory [?].

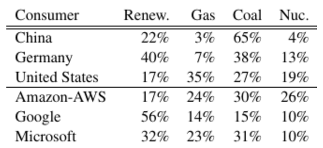
4. Inference Time Optimization

If inference time is taken to be a good proxy for computational expensiveness of a particular inference, there is a substantial body of research being done on model compression as a means of obtaining greater simplicity and efficiency within LLMs. Some methods of model compression include pruning, in which the model size is shrunk through the removal of parameters considered to be less relevant to accuracy, and quantization, in which the precision of the model weights and activation is reduced in order to minimize memory usage and accordingly increase the speed of inference [9]. Specifically, the techniques of mixed-precision quantization, post-training quantization, and quantized feature distillation when applied to model training are able to decrease both memory usage and resource consumption without sacrificing accuracy [9]. There are also CPU specific architecture optimizations, including the NIOT framework, which employs advanced cache management strategies, memory tiling, and thread allocation in order to reduce latency and thus inference time by almost 30% for the BERT model [9] When it comes to pruning, there are multiple approaches that can be taken to prune the model. Within the broader category of magnitude-based pruning, the goal is to get rid of parameters with small values because those values in theory contribute the least to the overall predictions of the models [9] In order to identify the "smallest" values, multiple different mathematical approaches can be taken based on the type of norm that is chosen to prune with. L1-Norm pruning, which utilizes the Manhattan distance, simpy eliminates all weights with the smallest L1 Norm [9]. On the other hand, L2 norm pruning uses the Euclidean distance ands ends up removing the smallest or least impactful filters, essentially getting rid of redundant units in the model [9]. Other broader categories of pruning include global and layer-wise pruning, which take different approaches towards shrinking the weight and parameter size, whether it be by layer, by location, by weight, or by some other mechanism [9]. By utilizing techniques such as quantization and pruning, we can reduce the computational intensity of some of the underlying mechanisms enabling LLMs.

5. Case Study: Training Stage

Researchers at UMass Amherst focused on the problem of measuring energy usage in the training stage. Specifically, they sample GPU and CPU power consumption from the NVIDIA System Management Interface and Intel's Running Average Power Limit interface while training a variety of models for max one day on an NVIDIA Titan X GPU [18]. The procedure used by the UMass Amherst researchers for calculating CO2e emissions during the training of deep neural networks involved first noting the total times required for training and the relevant hardware used from the papers behind the models being used. Then, the total power consumption is calculated via the equation $p_t = \frac{1.58t(p_c+p_r+gp_g)}{1000}$, where p_c represents the training-time average power draw from CPU sockets, p_r is that for main memory sockets, g is the number of GPUs needed to train, and p_g is the equivalent p for each GPU, with the constant term being the average 2018 Power Usage Effectiveness coefficient for data centers. Then, the estimated emissions are calculated via multiplying p_t by 0.954, the factor for determining CO2 given power consumed as established by the EPA, calculated by looking at the relative proportions of different energy sources [18].

Insights from this can be used to set the stage for how per-prompt emissions should be calculated and evaluated. One consideration is whether power is drawn from all three sources (CPU, main memory, and GPU) during prompt serving and what their respective relations are. Additionally, it may be worth investigating whether there is a more direct or precise way to determine emissions rather than relying on static, generalized constants, which are likely outdated and unlikely to be useful to generate a long-term model of energy consumption of ever-evolving models in a country with such a complex, nuanced grid. More broadly, it could be useful to conduct a sensitivity analysis into the final CO2e equation in order to determine how much the end



Consumer	Renew.	Gas	Coal	Nuc.
China	22%	3%	65%	4%
Germany	40%	7%	38%	13%
United States	17%	35%	27%	19%
Amazon-AWS	17%	24%	30%	26%
Google	56%	14%	15%	10%
Microsoft	32%	23%	31%	10%

Figure 1. Differing Energy Compositions of Nations and Companies

value fluctuates for small changes in each component value. With this context, we will be able to determine the most "important" variables to track with precision and formulate an updated equation for per-prompt emissions accordingly. Using the partial derivative method for estimating sensitivity, p_c and p_r are equally influential in the final calculation, with g and p_g's influence being dependent on one another. The most influential factor is, once again, time, upon which all of these partial calculations are based.

One adjustment that can be made is to the 0.954 value calculation, which currently is based on the relative proportions of different energy sources in the United States. For such an influential number in the final CO2e calculation, it is incredibly general and may not be the most applicable for each AI model, in part because different model-developers and companies are fueled by different energy compositions. The table below illustrates how the energy composition of the United States relates to those of top tech companies around 2017.

When analyzing the Transformer, ELMo, BERT, and GPT-2 models, the results are shown in the table extracted from the paper below [18].

One observation the researchers noted was that TPUs are more financially optimal for servicing workloads that apply to the relevant hardware such as BERT. This reinforces the core thesis in this paper that it is worthwhile to investigate how the workload or prompt type can relate to the energy consumption and possibly be optimized in that regard.

These characteristics contribute to the difficulties in optimizing for lower carbon footprints. This is because whereas traditional computing can be represented through a single, often straightforward metric such as end-to-end latency, each phase requires its own separate metric: time to first token in the prefill phase and time to generate each token in the decoding phase, with each phase having its own specifications involving compute, memory, latency and thus carbon footprint. For instance, the optimization strategies that befit compute-bound processes are not necessarily the most efficient memory-wise and vice versa, meaning that a single strategy cannot be cleanly applied to optimize the energy usage in both phases.

6. Case Study on Inference Stage

6.1. Tool Overview

While the prior section dove into calculations for energy usage during training time, it is important to also monitor the energy consumption associated with the actual usage of a model. To explore this, we will be utilizing the Chat UI - Energy tool developed by Hugging Face, a company providing platforms, datasets, infrastructure, and community for those exploring AI development [1]. This tool was built to estimate how much energy a user's conversations with their AI chatbot consumes. With a layout similar to typical AI chatbots, augmented with energy consumption and comparison metrics, and options to choose a model from a selection of LLMs including Qwen (three different versions), Llama 3.3, Gemma-3, Hermes-3, Mistral-Nemo, and Phi-3.5-mini, this tool allowed us to experiment with different prompts to different models, the results of which are described later in this section.

6.2. Tool Methodology

The model is given a system prompt, or a given set of instructions on how to generally behave in order to establish some level of context for the model, in this case, letting it know it is a tool to help users see how much energy usage. One key observation is that the user of the tool is able to see the system prompt.

This system prompt, in fact, is mutable by the user, and there is not a direct, precise set of formulas given for the energy usage numbers. The information the model is instructed to give, being specifically told it does not have access to the actual values, is that NVIDIA's NVML API is utilized on supported GPUs to calculate values. When this is not possible, values are estimated directly from inference time, with the estimated energy consumption being equivalent to the product of average power and inference team and an estimate of average power at 70W. This is also how the comparison to phone charges are calculated, with an estimate of a singular phone charge at 19 Wh or just under 70k joules. One key observation is that energy usage is not calculated directly but rather is the result of a series of operations performed on time elapsed in responding to a query, the metric the model actually has access to. In doing so, the model relies on the assumption that all computation being carried out at a given time is equivalent in nature and thus longer time being taken necessitates a more energy intensive operation.

6.3. Tool Results

Given that we want to investigate whether there are substantial differences in energy consumption along two different lines, along models and along prompts, our methodology is as follows. We have developed a set of three prompts

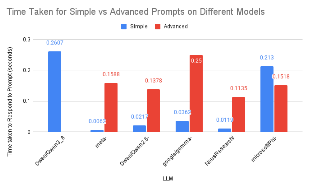


Figure 2. Time versus Model Experiment Results

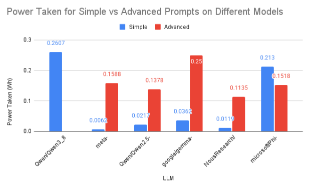


Figure 3. Power versus Model Experiment Results

designed to serve as representative samples for two different classes of inference tasks: straightforward answers and text generation with planning. We will enter these prompts into the Chat UI - Energy tool for different models and evaluate how the energy consumption statistics do or do not change. The prompts are shown below and henceforth referred to by their category of simple or advanced.

- Simple: What is the state capital of California?
- Generate a series of five limericks telling a children's story with a funny twist at the end.

These prompts were chosen in order to enable differentiation in model approaches: whereas the simple prompt requires straightforward regurgitation and/or pattern matching to answer, the advanced prompt requires generative capabilities and specifically hints at incorporating some form of long-term planning or chain-of-thought reasoning in order to develop multiple creative works with a coherent theme in which the later works need to be planned out ahead of time to allow for the setup of a twist at the end. N/A means that the model was unable (likely due to exceeding computational limitations) to generate the desired result and threw an error message instead. The charts below display results gleaned from this experiment. For the Qwen38B model, only results for the simple prompting scenario are displayed because the model timed out and returned an error on the advanced prompt.

Overall, we see that for most models the advanced prompt was more "difficult" to answer in that it required often an order of magnitude higher level of power usage and time taken to respond to. However, for one particular model, the Microsoft Phi model, the trend was actually reversed, with the advanced prompt taking slightly less time to service than the simple prompt. Additionally, we see that there is a nearly directly linear relationship between

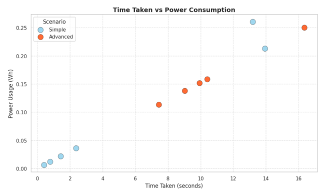


Figure 4. Time versus Power Experiment Results

Table 1. Comparison of model performance across simple and advanced prompts.

Model	Metric	Simple	Advanced
	Power usage (Wh)	0.2607	N/A
	% of phone charge	1.37	N/A
Qwen/Qwen3_8B	Time taken (s)	13.188	N/A
	Power usage (Wh)	0.0062	0.1588
	% of phone charge	0.03	0.84
meta-llama	Time taken (s)	0.403	10.393
	Power usage (Wh)	0.0217	0.1378
	% of phone charge	0.11	0.83
Qwen/Qwen2.5	Time taken (s)	1.418	9.019
	Power usage (Wh)	0.0362	0.25
	% of phone charge	0.19	1.32
google/gemma-3-27b-it	Time taken (s)	2.372	16.364
	Power usage (Wh)	0.0119	0.1135
	% of phone charge	0.06	0.60
Nous/Hermes-3	Time taken (s)	0.781	7.431
	Power usage (Wh)	0.213	0.1518
	% of phone charge	1.12	0.80
microsoft/Phi-3.5-mini	Time taken (s)	13.939	9.936

the time it takes a model to respond to a prompt and the amount of energy that same transaction requires.This follows from what the instructions tell us, that the formula for power usage is essentially a multiple of the measured time elapsed for the given prompt. Furthermore, we see that in almost all cases, there is a stark gap between both the time and power required to address simple prompts compared to advanced prompts. On top of that, it seems as though certain models are better suited for certain prompts and certain metrics. Qualitatively, while most models were straightforward in their responses, the Qwen response that timed out had a tendency to over-explain every step of its reasoning, even for the simplest task. Additionally, the Microsoft Phi model, likely motivated through a misinterpretation of its system prompt, took longer on the simple prompt because after answering the question it delved unprompted into its own energy usage. While this is still too little data to make any definitive conclusions, this exploration reveals the continued validity of our approach in believing different models and different prompts affect energy consumption and the environment in different ways, setting the stage for why different optimizations in different contexts may be needed.

7. Technical Deep Dive

7.1. Algorithm and Engineering Decisions

Given that different LLM designs and component algorithms can lead to different components of the overall model being prioritized at different times, it is reasonable to assume that different designs may lead to different computational intensities, a metric closely tied to a model's environmental impacts. To that end, this section will focus on methods, algorithms, and reasoning systems that are currently influencing the LLM landscape and poised to play a large role in the future. Specifically, we will look at the Transformer and Titan architectures; Chain-of-Thought reasoning; CPU, GPU, and TPU power consumption; and the von Neumann architectural system and alternatives.

7.2. Transformers

In order to further explore this idea that different LLM implementations may have different environmental effects when dealing with different types of problems, we will look at one particular component of LLMs, a network architecture called the Transformer that since its introduction via a Google research paper [19] in 2017, has become practically integral to modern LLM development. The Transformer is a network architecture entirely based on attention mechanisms, removing the need for convolutional or recurrent neural networks [19] First, we will take a look at what tasks Transformers are and are not optimized for. Then, we will look at how this intersects with the energy usage and inevitably environmental impact of Transformers and the LLMs they are part of. A key aspect of Transformer design is the self-attention mechanism, which was developed to aid in the identification of long-term dependencies within sequential data [9] Traditional models and alternatives to Transformers, including convolutional neural networks (CNNs), recurrent neual networks (RNNs), and long short-term memory networks (LSTMs), all are designed in a more sequential manner, making it difficult for them to pinpoint relationships spanning long ranges [9] [10]. A key reason for this is the propensity of these models to experience vanishing gradients within their internal calculations as a result of extremely small gradients being formed during the repeated backpropagation steps when dealing with long range data [9]. Transformers, on the other hand, gained popularity because of their ability to avoid the vanishing gradient issue and better understand relationships across ranges, making them especially useful for dealing with sequential data, specifically time series forecasting (TSF) problems. While these problems typically relied on LSTM and CNNs, the Transformer has quickly established itself as a strong competitor because it does not require the data to be dealt with sequentially, instead enabling the access of any part of the sequence through the self-attention mechanism [10]. Results of research applying the Transformer architecture to TSF problems included accuracy outperforming that of CNN and at par with that of LSTM. When it comes to training speed, Transformers are slower than CNNs but more temporally efficient than LSTM. However, Transformers often have a longer inference time than the other two systems [10]. Additionally, the self-attention mechanism characteristic of Transformers is an incredibly computationally intensive process, meaning that dealing with long time series' is incredibly computationally expensive [9]. The self-attention mechanism within a Transformer generally has a quadratic lower bound for both time and space with respect to the length of the input [8]. Given such a bound, one study has formulated an expression for energy consumption equal to $E = aCV^2fT$ with a being the activity factor, C the effective capacitance, V the supply voltage, f the operating frequency, and T the execution time. This formula renders it such that energy consumption is linearly proportional to execution time. Findings using this formula saw around 30% reduced inference times for models optimized via quantization and pruning, with not too significant corresponding drops in accuracy, with the implication that this would entail roughly 30% less energy consumption [9]. In order to maintain top of class accuracy, improvements in energy efficiency using these methods would top at slightly over 12% [9] As such, while Transformers are incredibly powerful tools, the resource-intensive nature of their characteristic self-attention mechanism leaves space for innovation and added efficiency.

7.3. Titans

Researchers at Google developed a novel neural network architecture called Titans, which gives models the ability to find and store tidbits of information during inference that could be useful in longer sequences . It does so by adding "neural memory" layers designed to grant models better flexibility in dealing with long and short term memory. In effect, this allows models to have fewer parameters than typical models for queries dealing with millions of tokens [5] [4]. Currently, large language models often have a Transformer architecture that uses the self-attention mechanism for relating tokens to one another but Titans does it differently. This allows for pattern matching in sequences of tokens. The issue is that as sequence lengths increase linearly, memory and thus computing costs associated with storing and calculating attention experience a quadratic increase [5] [4].

Methods that the Google researchers utilized to overcome these limitations included implementing "neural long-term memory" and "surprise." In neural long-term memory, rather than having information storage occur in the training stage, the neural memory contains a function enabling it to obtain new information during inference and accordingly adjust the memorization process, enabling greater generalization capabilities. The decision of what information chunks are worth remembering is aided by the notion of "surprise," in which sequences of tokens to memorize are prioritized based on how different they are from the kinds of knowledge already present in the existing weights and memory of the model [5] [4]. When looking at the theme of model-prompt alignment, that certain models are better equipped to take on certain tasks, the Titan architecture is no exception. Titans significantly outperforms other models, even those with multiple orders of magnitude more parameters, on tasks revolving around extricating signal from noise. In this way, Titans are a good fit for prompts asking the model to find and use bits of information entrenched in lengthy sequences of input or prompts asking the model to reason using facts embedded in disparate sections of long text [5] [4]. By clearly understanding which architectures are designed to best support which LLM-enabled task, model and application developers can ensure the most effective allocation of resources.

7.4. Chain-of-Thought Reasoning

When looking at how LLMs address the problem of reasoning, we generally take inspiration from our understanding of reasoning in humans. In particular, the idea of System 1 versus Sysem 2 reasoning has recently increased in importance. While System 1 reasoning represents forms of thinking that are intuitive, fast, and heavily reliant on heuristics and mental approximations, System 2 reasoning is inherently slower but more methodical and analytical. As of early 2025, the LLM landscape was dominated by incredibly powerful models that predominantly were limited to System 1 reasoning due to their focus on speed and prioritization of heuristics [12]. This falls in line with a popular view of LLMs and modern AI as arguably nothing more than incredibly advanced pattern-matching machines incapable of true thought. It also leads to the often amusing failures of certain models when prompted with seemingly easy problems or puzzles, often ones that require a clever yet obvious (to humans) trick or series thereof.

Recently, however, models including DeepSeek's R1 and OpenAI's o1 and o3 have been built with a slower, step-by-step reasoning system in mind, and it was found that rather than requiring large swaths of data to develop System 2 reasoning, it could be done with smaller datasets [12]. Many approaches have been explored to encourage slower and methodical reasoning in LLMs, including adding internal intermediate "pause" and "review" prompts, increasing the time allocated and required to generate a response, employing Chain-of-Thought reasoning, and using Monte Carlo Tree Search to explore the potentialities of different reasoning methods for a given problem [11]. Recently, Chain-of-Thought and Meta-Chain-of-Thought reasoning have become popular approaches used in LLMs to allow for more complex, less temporally linear tasks to be completed, such as those that require multi-stage planning before arriving at a final answer. However, there are scenarios where models trained using Chain-of-Thought reasoning are thought to have been deliberately deceitful [4]. As such, this system of reasoning, while enabling stronger responses to certain types of queries requiring long-term planning, brings with it another set of risks and uncertainties, ones that extend to its energy usage. As we saw in the test case study with HuggingFace's ChatUI tool, the model Qwen that employed a form of chain-of-thought reasoning as per its descriptions ended up overcomplicating a very simple request, jumping to justifications and conclusions that would seem unreasonable for a human. In doing so, it needlessly increased its own energy expenditure by overcomplicating its own task.

7.5. Processing Unit Allocation

Because GPUs can contain up to thousands of compute cores and thus enable high throughput and allow large amounts of computational power to be allocated to tasks, Convolutional Neural Networks (CNNs) are often trained via GPU computing [11]. Generally, GPUs are thought to speed up computationally intensive processes compared to CPUs. However, whether this speedup translates to different levels of energy consumption is a different question altogether. NVIDIA's cuDNN library, utilized by popular frameworks such as TensorFlow, enables deep learning accelerated by GPUs. Usage of cuDNN increases GPU power by an average of 16% and lowers energy consumption by an average of 42%. Although seemingly contradictory, this works because cuDNN utilizes GPUs at a higher rate, leading to higher power consumption, but the associated reduction in training time more strongly minimizes the total energy consumption. One notable finding is that, even when idle, the CPU uses a great deal of power, easily accounting for over 20% of total energy consumption [11]. To relate this back to our fundamental question of what architectural decisions affect model impacts on the environment, results from this study imply that a combined utilization of the CPU and GPU during the training phase is often best for CNN frameworks.

When looking at the needs of AI data centers compared to typical data centers, it is important to note that traditional data centers are centered around CPUs whereas data centers supporting AI applications such as LLMs involve the integration of GPUs and other units requiring higher server power to operate [20]. When looking at data center environmental impacts, however, we often need to take into account not only the total power needed to support the relevant computational processes but also the natural resources needed to enable the functioning of those computational processes. Specifically, we need to look at the cooling mechanism of these data centers. Whereas traditional data centers often made do with air cooling, data centers supporting AI applications such as LLMs, with their computationally intensive GPU technology, require increased power, resulting in the generation of increased heat and thus a demand for more advanced cooling which is often done through water: direct liquid cooling (DLC) [13]. Research has found that using DLC when dealing with AI workloads significantly improves GPU performance through increasing efficiency by roughly 3%, reducing power usage by over 10%, reducing execution times by over 5% and lowering computer chip temperatures by 20 degrees C [13]. Overall, the decisions on what type of processing unit to use different stages of LLM functionality heavily influence the energy and carbon footprint associated with those LLMs.

7.6. Von Neumann Architecture and Alternatives

Another aspect of LLM development that affects energy consumption actually relies outside of the language models entirely and deals with the computer architecture upon which these models are built, hosted, and run. In particular, we will dive into the von Neumann architecture, which was initially drafted in the mid 20th century and is still the most prominent architecture today. Within the von Neumann architecture, the memory and computing units (typically the CPU) are separate, allowing them to be designed, configured, and manufactured separately, enabling flexibility in selecting and pairing components for targeted applications [?]. However, this separation of the CPU from the memory can often be referred to as the von Neumann bottleneck, due to how it inherently limits the execution rate of instructions [?]. Whereas processing and memory have undergone large increases in efficiency over the past decade, data transfer efficiency has not had the same trend. Thus, current computer performance, especially when dealing with compute and memory-intensive applications such as LLMs, is often restricted by the data transfer rate between the different components: compute and memory [?]. In fact, AI applications are a sort of double-whammy on the efficiency of von Neumann architecture compared to traditional computing. On the one hand, when dealing with AI applications such as LLMs, there is a higher amount of data to move (billions of model parameters and weights), and on the other hand, the data needs to move further between memory and compute processing locations [?]. This doubly impacts energy consumption because there are both higher numbers of energy-requiring transactions given the influx of memory data needing to be moved as well as a greater energy expenditure needed to move the memory further distances. This is because greater distance to the processor from the memory requires that increased energy is used in moving the memory. In order to understand why, we look at the fundamental physics of electronics, as we can imagine the system being one where a copper wire transmits a 1 by being charged and a 0 by being discharged. Because the energy needed to charge and discharge wires increases proportionally to the wire's length, long wires require more energy to transmit the same information. The reason we have greater distances between memory and processing in LLM computing is because the large amount of model weights needed requires a great deal of storage, and a physically larger storage unit comes with more distance between the storage and the memory [?]. Given how the application of LLM computing dampens the efficiency of von Neumann architecture in two mutually exacerbating ways, there is a great deal of research being done on alternative architectures more amenable to the high data and high throughput requirements of LLM serving.

8. Conclusion and Toolkit Formation

The motivation for developing a conceptual toolkit guiding the systematic yet dynamic evaluation of the environmental impacts associated with different LLMs lies in the understanding gleaned from prior sections of just how rapidly this industry and its underlying technologies is developing and how inherently complex it is to determine and optimize energy usage and emissions reductions within these complicated systems. Insights from evaluating different estimation methodologies used currently reveal that many of these energy consumption estimates rely on very broad assumptions, are often incredibly volatile, and are often dependent on particular variables that may be subject to change as LLMs develop. Thus, rather than provide a definitive equation or series of equations to calculate carbon emissions for a singular model, this toolkit will identify factors and variables associated with LLM development that have direct effects on the environment, situate those factors in the context of how they may be poised to change in the future and how sensitive the net environmental impact will be on each of those factors, and broadly identify aspects of development to account for in developing these numbers.

Over the course of this paper, we have seen that a multitude of interconnected factors affect the overall energy consumption and thus environmental impact of LLMs. Even the very act of measuring any or all of these factors and obtaining consistent and reliable metrics by which to evaluate environmental impacts is difficult and often obfuscated in various ways. Oftentimes, even if the specific energy consumption can be measured for a particular model, factor, or aspect of models in theory, it is difficult to amalgamate that with the measurements for the numerous other components making up not only a singular LLM but the entire LLM landscape today. As such, various heuristics are often used. Within the realm of training and inference-time environmental analysis, we often see the use of energy as a proxy for environmental impact, computational complexity as a proxy for energy use, and training/inference time as a proxy for computational complexity. While these simplifications can yield us general orders of magnitude or allow us a basis by which we can compare different models and systems, they require a great deal of simplifications that impact their ability to be precise determiners of environmental impact.

Even though the factors themselves are dynamically changing as the field evolves, we can identify a set of factor categories that influence the environmental effects of the broader LLM systems. To recap these we will traverse them from the highest to lowest levels of abstraction.

At the highest level, we can identify that the facets of environmental impact that we need to pay attention to include not only the energy needed for the electricity powering LLMs but also resource usage that supports the broader LLM infrastructure. These could include both the embodied energy and emissions that go into the manufacturing and production of the electronic equipment, which we did not explore in this particular paper, and the water usage needed to enable the functionality of the data centers powering AI applications such as LLMs, which we explored within this paper. Additionally, this level encapsulates various scope 2 and 3 emissions associated with the companies developing these models and their associated environmental impacts.

When taking a step forward into the model level, motivated by historical context on impact differentiation within different use cases of the same technologies, we see broadly that different models have differing emissions profiles, and some models may outperform others in terms of accuracy or energy efficiency when it comes to different tasks. This, combined with the complexity of optimizing the entirety of the LLM pipeline given often contradicting constraints for different facets of the models, builds up an exigence for the development of use-case specific models as opposed to general models, which currently dominate the LLM research landscape. With specific models, there is increased room to optimize at the lower levels such as implementing particular architectures and variations that align with efficient computations of certain types.

When looking at an even lower level, we see how individual algorithms and architectural decisions affect the overall energy consumption of LLMs. This can include how to reason about optimization in compute versus memory bound scenarios; how to allocate compute to different processing units (CPUs, GPUs, and TPUs) at train versus test time; whether to follow von Neumann architecture; whether to employ Transformers, Titans, or other architectures, and if so, how; what reasoning methods to implement and prioritize; and how to mathematically build out and shrink the algorithms that carry out the large numbers of computations required such that model complexity is minimized without sacrificing accuracy. By paying close attention to these facets not just individually but in their interactions with one another, we will be able to develop a more accurate, more holistic, and more scalable infrastructure for reasoning about how rapidly developing LLMs may be affecting the environment.

References

[1] Chat UI Energy Score - a Hugging Face Space by jdelavande. 6

[2] L. Badea and M. C. Mungiu-Pupzan. The Economic and Environmental Impact of Bitcoin. *IEEE Access*, 9:48091–48104, 2021. 3

[3] E. M. Bender, T. Gebru, A. McMillan-Major, and S. Shmitchell. On the Dangers of Stochastic Parrots: Can Language Models Be Too Big? . In *Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency*, pages 610–623, Virtual Event Canada, Mar. 2021. ACM. 4

[4] Y. Chen, J. Benton, A. Radhakrishnan, J. Uesato, C. Denison, J. Schulman, A. Somani, P. Hase, M. Wagner, F. Roger, V. Mikulik, S. Bowman, J. Leike, J. Kaplan, and E. Perez. Reasoning Models Don't Always Say What They Think. 8, 9

[5] B. Dickson. Google's new neural-net LLM architecture separates memory components to control exploding costs of capacity and compute, Jan. 2025. 8, 9

[6] Y. Ding and T. Shi. Sustainable LLM Serving: Environmental Implications, Challenges, and Opportunities : Invited Paper. In *2024 IEEE 15th International Green and Sustainable Computing Conference (IGSC)*, pages 37–38, Nov. 2024. ISSN: 2993-2084. 4

[7] S. Griffiths. Why your internet habits are not as clean as you think, Mar. 2020. 2, 3

[8] F. D. Keles, P. M. Wijewardena, and C. Hegde. On The Computational Complexity of Self-Attention, Sept. 2022. arXiv:2209.04881 [cs]. 8

[9] A. Kermani, E. Zeraatkar, and H. Irani. Energy-Efficient Transformer Inference: Optimization Strategies for Time Series Classification. *International Journal of Computer Applications*, 186. 5, 8

[10] P. Lara-Benítez, L. Gallego-Ledesma, M. Carranza-García, and J. M. Luna-Romera. Evaluation of the Transformer Architecture for Univariate Time Series Forecasting. In E. Alba, G. Luque, F. Chicano, C. Cotta, D. Camacho, M. Ojeda-Aciego, S. Montes, A. Troncoso, J. Riquelme, and R. Gil-Merino, editors, *Advances in Artificial Intelligence*, pages 106–115, Cham, 2021. Springer International Publishing. 8

[11] D. Li, X. Chen, M. Becchi, and Z. Zong. Evaluating the Energy Efficiency of Deep Convolutional Neural Networks on CPUs and GPUs. In *2016 IEEE International Conferences on Big Data and Cloud Computing (BDCloud), Social Computing and Networking (SocialCom), Sustainable Computing*

and Communications (SustainCom) (BDCloud-SocialCom-SustainCom), pages 477–484, Oct. 2016. 9

[12] Z.-Z. Li, D. Zhang, M.-L. Zhang, J. Zhang, Z. Liu, Y. Yao, H. Xu, J. Zheng, P.-J. Wang, X. Chen, Y. Zhang, F. Yin, J. Dong, Z. Li, B.-L. Bi, L.-R. Mei, J. Fang, Z. Guo, L. Song, and C.-L. Liu. From System 1 to System 2: A Survey of Reasoning Large Language Models, Apr. 2025. arXiv:2502.17419 [cs]. 9

[13] B. Ramakrishnan, C. Turner, H. Alissa, D. Trieu, F. Rivera, L. Melton, M. Rao, S. Chigullapalli, T. Getachew, V. Prodanovic, R. Lankston, C. Belady, and V. Oruganti. Understanding the Impact of Data Center Liquid Cooling on Energy and Performance of Machine Learning and Artificial Intelligence Workloads. *Journal of Electronic Packaging*, 147(021003), Dec. 2024. 10

[14] G. Rosalsky. Why the AI world is suddenly obsessed with a 160-year-old economics paradox. *NPR*, Feb. 2025. 4

[15] A. Shehabi, S. Smith, D. Sartor, R. Brown, M. Herrlin, J. Koomey, E. Masanet, N. Horner, I. Azevedo, and W. Lintner. United States Data Center Energy Usage Report. Technical Report LBNL–1005775, 1372902, June 2016. 2

[16] A. Singh, N. P. Patel, A. Ehtesham, S. Kumar, and T. T. Khoei. A Survey of Sustainability in Large Language Models: Applications, Economics, and Challenges. In *2025 IEEE 15th Annual Computing and Communication Workshop and Conference (CCWC)*, pages 00008–00014, Jan. 2025. 4

[17] C. Stoll, L. Klaaßen, and U. Gallersdörfer. The Carbon Footprint of Bitcoin. *Joule*, 3(7):1647–1661, July 2019. 3

[18] E. Strubell, A. Ganesh, and A. McCallum. Energy and Policy Considerations for Deep Learning in NLP, June 2019. arXiv:1906.02243 [cs]. 5, 6

[19] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin. Attention Is All You Need, June 2017. 8

[20] Y. Wang, Y. Han, K. Han, and J. Shen. Does DeepSeek curb the surge of energy consumption in data centers? *The Innovation*, page 100944, May 2025. 4, 9

[21] D. Zipper. What a 160-year-old theory about coal predicts about our self-driving future, Sept. 2024. 4