Comparative Evaluation of Energy Efficiency in Large Language Models: Analyzing Improvements Across Incremental Versions in Inference Tasks

Andrei Dragomir
2669304
VU Amsterdam
a.dragomir@student.vu.nl

Adam Bouafia
2856516
VU Amsterdam
a.bouafia@student.vu.nl

Vlad-Andrei Serghei
2850208
VU Amsterdam
v.serghei@student.vu.nl

Nahid Jahanianarangeh
2859504
VU Amsterdam
n.jahanianarangeh@student.vu.nl

Punya Kapoor
2758893
VU Amsterdam
p.kapoor@student.vu.nl

Younes Boubbou
2819754
VU Amsterdam
y.boubbou@student.vu.nl

ABSTRACT

Timelog Link: Here
Context. ✎ at the end
Goal. ✎ at the end
Method. ✎ at the end
Results. ✎ at the end
Conclusions. ✎ at the end

ACM Reference Format:
Andrei Dragomir, Adam Bouafia, Vlad-Andrei Serghei, Nahid Jahanianarangeh, Punya Kapoor, and Younes Boubbou. 2020. Comparative Evaluation of Energy Efficiency in Large Language Models: Analyzing Improvements Across Incremental Versions in Inference Tasks . In . ACM, New York, NY, USA, 13 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

Large Language Models (LLMs) are neural network architectures with billions of parameters, trained on vast datasets to perform a wide range of natural language processing (NLP) tasks such as text generation, translation, and question answering [1].

These models are typically based on transformer architectures and have proven to be powerful in generating human-like responses across various domains [2].

LLMs such as ChatGPT, Gemini, Copilot, and LLama have gained widespread popularity in recent years, both in the public domain as virtual assistants and in the development of NLP systems for enterprise applications. For example, ChatGPT reached 1 million visitors within 5 days and has over 100 million visitors each month [3]. The LLM market is seeing constant growth and it is estimated that by 2030 it will reach a $259,8 million value from the $1,590 million it had in 2023 [4] [5].

Conference'17, July 2017, Washington, DC, USA

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM
https://doi.org/10.1145/nnnnnnn.nnnnnnn

Two prominent examples of LLMs are **ChatGPT** (Generative Pre-trained Transformer), created by OpenAI, and **LLaMA** (Large Language Model Meta AI), developed by Meta [1, 6].

ChatGPT, which has been released in multiple versions, showcases an impressive ability to generate contextually relevant, human-like responses across a wide range of topics, making it popular for various conversational AI applications.

LLaMA, on the other hand, aims to deliver performance comparable to other leading models while being optimized for resource efficiency [6].

Although these models differ in architecture, training data, and implementation, both represent significant advancements in NLP technology. Understanding how these models evolved across different iterations is crucial to evaluating their environmental impact, as the demand for state-of-the-art LLMs continues to rise [7].

The power and flexibility of LLMs come at the cost of substantial computational resources, which leads to high energy consumption [8]. This raises questions about the environmental sustainability of LLMs.

For example, some estimates suggest that ChatGPT alone emits around 8.4 tons of CO2 annually during operation [7]. The extent of these emissions depends heavily on the energy sources used by the data centers, with those relying on fossil fuels contributing to even higher CO2 output [9, 10].

Additionally, data centers require large quantities of water for cooling purposes. The training of GPT-3, for instance, is estimated to have consumed about 700,000 liters of water [11].

Optimizing the energy and resource consumption of LLMs is a critical step in reducing their environmental footprint.

Techniques such as model quantization and fine-tuning can improve computational efficiency, allowing smaller, specialized models to achieve high performance with lower energy and resource consumption compared to general-purpose models like ChatGPT [12].

This paper aims to evaluate the energy efficiency of LLMs across different versions by examining tasks such as text generation, question answering, and summarization.

By analyzing how energy consumption has changed across successive model versions, we aim to gain insights into the

trade-offs between energy efficiency and performance, ultimately highlighting the importance of energy-efficient design in the further development of LLMs.

2 RELATED WORK

Lacoste et al. [13] provide a comprehensive framework for assessing the carbon emissions produced during the training of machine learning models.

They estimate that training a large-scale model like GPT-3 can generate up to 552 metric tons of CO2, which is roughly equivalent to the carbon footprint of 120 gasoline-powered vehicles driven for an entire year.

The authors identify crucial factors affecting emissions, including server location, hardware specifications, and the duration of the training process. To address these concerns, they introduced the Machine Learning Emissions Calculator, a tool designed to estimate and mitigate the environmental impact associated with model training.

While Lacoste et al.'s work focuses on training, our study evaluates the energy efficiency of LLMs during the inference phase, which is crucial as models are increasingly deployed at scale.

Unlike training, inference is an ongoing process that contributes significantly to the overall environmental impact every time the model is used.

Argerich et al. [14] introduce EnergyMeter, a Python-based tool developed to monitor the energy consumption of LLMs during inference tasks.

Their experiments reveal that larger models, such as GPT-3, can consume up to 50 watts per second during inference, depending on task complexity and model size.

Additionally, their study highlights that while increasing the batch size can improve inference efficiency by up to 30%, the relationship between energy consumption and inference latency is complex.

Larger models, although more power-hungry, tend to perform faster inference due to parallelization capabilities, leading to a trade-off between energy use and performance.

Our study complements theirs by bench-marking energy efficiency across different versions of LLMs, tracking improvements (or regressions) in energy use across successive model iterations.

Lin and Voas [15] focus on strategies to lower the energy consumption of LLMs through fine-tuning pre-trained models and designing new models with lower energy demands.

They estimate that fine-tuning pre-trained models can reduce energy consumption by up to 50%, particularly for domain-specific applications.

Moreover, the authors discuss MIT's Learned Linear Growth Operator (LiGO), which enables efficient scaling of smaller models into larger ones.

While their study centers around optimizing LLMs during training, our research focuses on evaluating energy consumption during inference, bridging the gap between training-focused optimizations and inference-heavy applications like conversational agents.

Samsi et al. [16] conducted a detailed analysis of the energy costs associated with inference in LLMs, focusing on models like Meta's LLaMA.

They found that the 65B-parameter LLaMA model consumed approximately 3 kWh of electricity over 24 hours of continuous inference, compared to 1 kWh for the 7B model.

This non-linear scaling in energy consumption, particularly with larger models, reflects the complex trade-offs between model size, energy use, and inference speed.

Samsi et al. also explored how batch sizes and model sharding affect energy efficiency, concluding that larger batch sizes lead to better energy utilization, though the number of GPUs involved plays a significant role.

Our study builds on these findings by comparing the energy consumption of different LLM versions and evaluating whether newer versions provide improved energy efficiency.

Xu et al. [17] investigate the energy efficiency of training neural network architectures, particularly convolutional neural networks (CNNs). Their empirical study found that energy efficiency can vary by as much as 40% depending on hardware configurations and optimization techniques. Although their study focuses on training, the identified factors—such as hardware utilization and model complexity—are also directly relevant to inference tasks.

Our research applies similar energy measurement methodologies to assess energy consumption during the inference phase.

Specifically, we use tools such as Experiment Runner and Powerstat to benchmark the energy efficiency of different LLM versions.

In summary, the above studies focus primarily on either training-phase emissions or optimizing energy efficiency during inference. Our study contributes to this ongoing research by specifically bench-marking and comparing energy efficiency across successive LLM versions during inference.

This approach provides insights into whether newer versions of LLMs are becoming more energy-efficient as they grow in complexity and capability.

3 EXPERIMENT DEFINITION

In this section, we use the GQM framework [18] to further define the experiment. Therefore, our goal is formulated as defined in Table 1.

Analyze	incremental releases of LLMs
for the purpose of	evaluation
with respect to their	energy efficiency and performance
from the point of view of	model developers and software developers
in the context of	different inference task types

Table 1: Goal Definition

3.1 Goal

The primary goal of our experiment, as expressed in Figure 1, is focused on evaluating three large language models developed by leading industry organizations in order to conclude whether or not newer models present optimizations not only on their capabilities but also in their energy efficiency and resource utilization when compared to their predecessors in the context of different inference task types.

In addition to evaluating model capabilities, this study is relevant for both model developers who are concerned with the operational costs of using LLMs at scale as well as software developers that would want to make use of these models within their projects. Energy consumption is directly linked to infrastructure expenses, especially in cloud-based environments. By assessing the energy efficiency of different models, developers can make informed choices that optimize both performance and cost. Moreover, understanding how newer models perform in terms of energy use is essential for ensuring scalability without disproportionately increasing resource consumption.

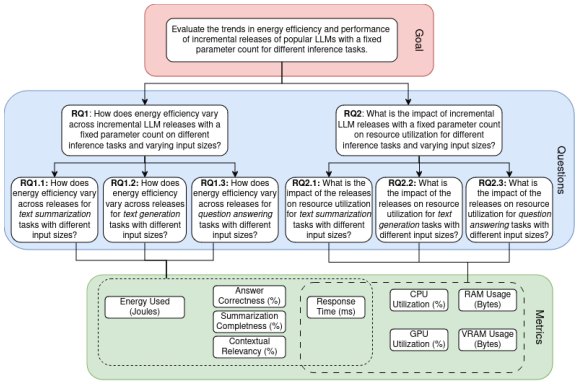

Figure 1: GQM Diagram

3.2 Research Questions

With the previously described goal in mind, we formulated the following research questions:

- **RQ1:** How does energy efficiency vary across incremental LLM releases with a fixed parameter count on different inference tasks and varying input sizes?
  - **RQ1.1:** How does energy efficiency vary across releases for text summarization tasks with different input sizes?
  - **RQ1.2:** How does energy efficiency vary across releases for text generation tasks with different input sizes?
  - **RQ1.3:** How does energy efficiency vary across releases for question answering tasks with different input sizes?
- **RQ2:** What is the impact of incremental LLM releases with a fixed parameter count on resource utilization for different inference tasks and varying input sizes?
  - **RQ2.1:** What is the impact of the releases on resource utilization for text summarization tasks with different input sizes?
  - **RQ2.2:** What is the impact of the releases on resource utilization for text generation tasks with different input sizes?
  - **RQ2.3:** What is the impact of the releases on resource utilization for question answering tasks with different input sizes?

With these in mind, we can formulate a brief description of what each main research question (RQ1 and RQ2) would bring out about the LLMs tested:

RQ1 examines how energy consumption is affected by incremental LLM updates while maintaining the same number of model parameters, across different input sizes. By focusing on energy efficiency, our experiment aims to quantify the amount of energy required for each model to complete tasks like text generation, summarization, and question answering, with various input lengths. This will reveal whether newer versions of LLMs use energy more or less efficiently as they evolve, and how input size influences energy consumption, even when the models have equal complexity in terms of parameter count.

On the other hand, RQ2 focuses on how different LLM versions use computational resources—such as CPU, GPU, and memory—while performing various inference tasks with varying input sizes. The aim is to understand whether newer LLM releases are optimized in terms of resource allocation when completing the same tasks (e.g., text generation, summarization, QA) across different input sizes. The experiment will reveal the impact of improvements or changes to LLM architectures on system resource demands, and how input length factors into resource usage, which is crucial for deploying these models in cost-efficient and scalable environments.

Lastly, this selection of tasks is based on the fact that text generation, question answering, and text summarization are the most common NLP tasks in the day-to-day usage of Large Language Models (LLMs) [19].

3.3 Metrics

In order to formulate answers to the above-stated research questions, we will have to use different tools in order to extract the following metrics:

Energy Used (in Joules): This metric measures the total energy consumption during each inference task performed by the LLM. It is used to reflect the energy efficiency of the

model, helping to understand how much energy is required to complete tasks like text generation, summarization, or question answering. Lower energy use with the same or better performance indicates improved energy efficiency.

Model Performance:

- **Answer Correctness (0-1 Score):** This metric assesses the accuracy of the model's response to question-answering tasks, both extractive and generative. It is based on how correct or appropriate the answer is relative to the ground truth. A score closer to 1 signifies a highly accurate response, while a score closer to 0 indicates poor performance.
- **Summarization Completeness (0-1 Score):** This score evaluates how well the LLM condenses the original text while retaining the essential information. A score of 1 implies the summary contains all critical details, while a score of 0 suggests significant information loss.
- **Contextual Relevancy (0-1 Score):** This metric checks how contextually appropriate the LLM's output is in relation to the provided input. It applies to both paraphrasing and auto-completion tasks, where the generated text should align with the input context. A higher score indicates better contextual relevancy.

Response Time (in milliseconds): This metric captures the latency of the model, i.e., the time taken by the LLM to generate an output from the moment it receives a prompt. Faster response times (lower values) are generally preferable for real-time applications.

GPU Utilization (percentage): This metric indicates the percentage of the available GPU capacity being used by the LLM during inference. It helps assess how much of the GPU's power is being consumed, with higher percentages suggesting greater usage.

CPU Utilization (percentage): Similar to GPU utilization, this metric reflects how much of the available CPU capacity is used by the model while performing tasks. High CPU usage could indicate resource inefficiency or intensive computation requirements.

Memory Usage (in bytes): This metric tracks how much memory (RAM and VRAM) the LLM consumes while performing tasks. Efficient models will use less memory to achieve the same or better performance, which is important for scalability, especially when running multiple models simultaneously or deploying in memory-constrained environments.

4 EXPERIMENT PLANNING

4.1 Subjects Selection

Our goal in this experiment is to evaluate different LLM candidates[1] for which we have found incremental versions in order to compare their performance.

Different parameter sizes (1.5B, 7B, 13B, etc.) have a direct effect on the baseline energy consumption of the model. Hence, we need to choose models for which incremental versions have a consistent parameter size to maintain the validity of the experiment.

Given the limitations of our testing platform, we are inclined to choose models with the smallest yet shared amount of parameters possible. With that in mind, we have taken into consideration three LLM candidates, namely:

- Alibaba Cloud's Qwen
- Google's Gemma
- MistralAI's Mistral

This candidate selection is mostly influenced by the popularity of the models selected as well as their open-source nature.

Beginning with the Qwen model, we looked over the Qwen 1(2024), 1.5(2024), 2(2024), and 2.5(2024) versions, which all have a 7B parameter size.The Qwen GitHub repository has 30 contributors and has been starred by almost 15000 people. In the industry it has been integrated by Xiaomi into their AI assistant providing new features for their smartphones and smart electric vehicles [20].

Next, MistralAI's Mistral models Mistral-7B-v0.1(2023), Mistral-7B-v0.2(2024), and Mistral-7B-v0.3 (2024) have a fixed size of 7B parameters across the different versions. MistralAI has started a collaboration with Microsoft, integrating it into AWS bolstering both the model's potential with the AzureAI infrastructure and the platform's features with everything that Mistral has to offer [21].

Lastly, we checked out Gemma, taking into consideration as candidates the Gemma 1(2024), Gemma 1.1(2024), and Gemma 2(2024), all having a consistent size of 2B parameters. Gemma is a lightweight model that can easily be used to create chatbots or virtual assistants as it is optimized for efficiency while locally deployed [22].

With that being said, table 2 shows the best-fit parameter size for our context as well as the available versions found for each one of the candidates.

Candidate	Versions	Parameter Size
Qwen	v1, v1.5, v2, v2.5	7 Billion
Gemma	v1, v1.1, v2	2 Billion
Mistral	v0.1, v0.2, v0.3	7 Billion

Table 2: Candidate Sets

One thing to note for the Mistral and Gemma candidates is that the selected versions sets are all instruct versions of the model but this does not affect our work as it is not the point of this paper to draw any comparative conclusions on the performance of the models between different LLM candidates.

In conclusion, while all the above-described options have different abilities and limitations, they all are characterized by an equal parameter size across their different versions making them strong candidates for our study while also meeting the requirements of our testing environment.

We also considered Meta's LLama model as a candidate but we only found a shared parameter size of 70 Billion parameters which, due to the nature of our testing platform does not fit within this experiment.

[1]We consider an LLM candidate as the family of incremental LLM versions selected for a specific model.

4.2 Experimental Variables

4.2.1 Independent Variables. The independent variables are factors that are systematically manipulated during the experiment to evaluate their impact on performance and energy efficiency. They include:

- **LLM Group (Blocking Factor):** The LLM models are divided into groups, with each group representing a distinct LLM candidate.
- **LLM Version (Main Factor):** Within each LLM group, different versions are compared to assess how incremental releases of the models affect performance and energy efficiency. A total of 10 versions are distributed across the three groups:
  - 4 versions of Qwen
  - 3 versions of Gemma
  - 3 versions of Mistral
- **Task Type (Blocking Factor):** The experiment is also blocked by task type, ensuring that comparisons within each task type are made across versions of the LLMs:
  - Text Generation (Paraphrasing)
  - Question Answering (Extractive QA)
  - Text Summarization (Extractive Summarization)
- **Input Size (Co-factor):** The input size is varied across two categories to assess its impact as a secondary factor influencing performance and energy efficiency across different versions and task types:
  - Short input
  - Long input

4.2.2 Dependent Variables. The dependent variables are the outcome measures that will be collected to evaluate the impact of the independent variables. These include:

- **Energy Used (in Joules)**: The total energy consumption during inference for each LLM version performing a task of a specific input size.
- **Model Performance Score**: Evaluating the model's ability to perform a task. Different scoring methods are used for each task type as seen in table 3. More on the scoring methods can be found in Section 5 of this document.
- **Response Time (in milliseconds)**: The time taken for each LLM version to produce an output in response to a given prompt.
- **CPU Utilization (in percentages)**: Average CPU utilization of the LLM while performing a task.
- **GPU Utilization (in percentages)**: Average GPU utilization of the LLM while performing a task.
- **RAM Usage (in Bytes)**: Average RAM allocated for the LLM while performing a task.
- **VRAM Usage (in Bytes)**: Average VRAM memory allocated for the LLM while performing a task.

Task Type	Assessment Method
Text Generation	Contextual Relevancy
Question Answering	Answer Correctness
Text Summarization	Summarization Completeness

Table 3: Task - Assessment Pairs

The dependent variables are going to be used to answer different research questions and the question to variable dependencies are described in table 4

Variable	Questions
Energy Used	RQ1.1, RQ1.2, RQ1.3
Model Perform. Score	RQ1.1, RQ1.2, RQ1.3
Resource Utilization	RQ2.1, RQ2.2, RQ2.3
Response Time	RQ1.1, RQ1.2, RQ1.3, RQ2.1, RQ2.2, RQ2.3

Table 4: Variable - Questions Pairs

4.2.3 Control Variables. Control variables are those factors that remain constant throughout the experiment to ensure valid comparisons between different runs. These include:

- **Model Parameter Count**: The parameter size for each LLM candidate will remain constant across incremental releases to ensure fair comparisons of energy efficiency and resource utilization.
- **Hardware Environment**: All LLM versions will be run on the same hardware configuration (CPU, GPU and memory) to prevent variability in resource utilization caused by hardware differences.
- **Task Prompt Structure**: The structure and format of prompts for each task type (e.g., question answering, summarization) will be kept consistent across all input sizes and versions of LLMs to ensure that input complexity is systematically controlled.
- **Evaluation Metric Thresholds**: The thresholds for model performance metrics (e.g., correctness, completeness, relevancy) will remain constant for all versions and task types to provide a uniform basis for comparison when evaluating resource utilization.

4.3 Experimental Hypotheses

The objective of this experiment is to investigate the energy consumption and resource utilization of Large Language Models (LLMs) during the inference phase as they evolve across different versions. The focus is on whether newer versions of LLMs are more energy-efficient and utilize resources (such as CPU, GPU and memory) more effectively compared to their predecessors. The following hypotheses guide the experiment:

4.3.1 ***RQ1 and subsequent sub-RQs.*** Null Hypothesis H0: There is no significant difference in energy efficiency between LLM releases with a fixed parameter count on different tasks across different input sizes.

$$H_1 : E_i = E_j$$

where E_i and E_j represent the energy efficiency during inference for different versions of the LLM ($i, j \in \{1, 2, 3, 4\}$) , for the same task.

Consequently, the null hypotheses for the sub-RQs will be defined as:

- H0 1.1: There is no significant difference in energy between LLM releases for text summarization tasks across different input sizes.

- H0 1.2: There is no significant difference in energy efficiency between LLM releases for text generation tasks across different input sizes.
- H0 1.3: There is no significant difference in energy efficiency between LLm releases for question answering tasks across different input sizes.

Alternative Hypothesis HA: There is a significant difference in energy efficiency between LLM releases with a fixed parameter count on different tasks across different input sizes.

$$H_1 : E_{\text{i}} \neq E_{\text{j}}$$

where E_{new} and E_{old} represent the energy efficiency during inference for different versions of the LLM ($i, j \in \{1, 2, 3, 4\}$) , for the same task. Here again, the hypothesis can be applied to the sub-RQs:

- HA 1.1: There is a significant difference in energy between LLM releases for text summarization tasks across different input sizes.
- HA 1.2: There is a significant difference in energy efficiency between LLM releases for text generation tasks across different input sizes.
- HA 1.3: There is a significant difference in energy efficiency between LLM releases for question answering tasks across different input sizes.

4.3.2 ***RQ2 and subsequent RQs.*** Since we have multiple types of resources used, we can define separate null-alternative hypotheses for each type of resource. The general null hypothesis, though, would be as follows:

$$H_3 : R_{\text{i}}(S) = R_{\text{i+1}}(S)$$

and the alternative hypothesis would be:

$$H_3 : R_{\text{i}}(S) \neq R_{\text{i+1}}(S)$$

where $R_{\text{i}}(S)$ and $R_{\text{i+1}}(S)$ represent the resource utilization (such as CPU, GPU or memory) during inference for different versions of the LLM ($i, j \in \{1, 2, 3, 4\}$) with the same parameter count, across different input sizes S.

Consequently, we can define the null-alternative hypotheses for the sub-RQs:

- H0: There is no significant difference in **CPU utilization** between LLM versions for text summarization, text generation, or question answering tasks across different input sizes ($\text{CPU}_i = \text{CPU}_j$ for some $i, j \in \{1, 2, 3, 4\}$).
- Ha: There is a significant difference in **CPU utilization** between at least one LLM version for text summarization, generation, or question answering tasks across different input sizes ($\text{CPU}_i \neq \text{CPU}_j$ for some $i, j \in \{1, 2, 3, 4\}$).
- H0: There is no significant difference in **GPU utilization** between LLM versions for text summarization, generation, or question answering tasks across different input sizes ($\text{GPU}_i = \text{GPU}_j$ for some $i, j \in \{1, 2, 3, 4\}$).
- Ha: There is a significant difference in **GPU utilization** between at least one LLM version for text summarization, generation, or question answering tasks across different input sizes ($\text{GPU}_i \neq \text{GPU}_j$ for some $i, j \in \{1, 2, 3, 4\}$).
- H0: There is no significant difference in **Memory utilization** between LLM versions for text summarization, generation, or question answering tasks across different input sizes ($\text{Mem}_i \neq \text{Mem}_j$ for some $i, j \in \{1, 2, 3, 4\}$).
- Ha: There is a significant difference in **Memory utilization** between at least one LLM version for text summarization, generation, or question answering tasks across different input sizes ($\text{Mem}_i \neq \text{Mem}_j$ for some $i, j \in \{1, 2, 3, 4\}$).

4.4 Experimental Design

The experiment follows a **two-factor factorial design** with blocking on the LLM candidate sets as well as the task types. The primary goal is to evaluate the energy efficiency and resource utilization of different LLM releases while systematically testing the effects of newer model versions as well as input size on model performance.

The types of tasks and input sizes that will be tested in our experiment are as follows:

- **Text Generation:** Generating coherent and contextually relevant text based on a prompt. In our experiment, we will focus on the paraphrasing aspect of text generation tasks: Completing text based on the context and relevance.
  - **Short Input:** A single sentence, such as "The weather today is..."
  - **Long Input:** Multiple paragraphs of text, describing the same thing but with more detail.
- **Question Answering (QA):** Extracting relevant information or generating responses based on a given question. In our experiment, we will focus on the extractive aspect of question answering tasks: The model selects a span of text from a provided document that answers a question.
  - **Short Input:** A single sentence or short passage (e.g., "What is the capital of France?")
  - **Long Input:** Multi-paragraph texts with multiple informative points.
- **Text Summarization:** Condensing a lengthy piece of text into a shorter, more concise form while retaining the most important information. In our experiment, we will focus on the extractive summarization aspect of text summarization tasks: Selecting and extracting key sentences or phrases from a provided text.
  - **Short Input:** A single paragraph with a few key points.
  - **Long Input:** Multi-paragraph texts.

4.4.1 Independent Variables to Factors. To account for the variability between different LLM architectures, we employ blocking by LLM candidate sets. The experiment includes three LLM candidates—Qwen, Gemma and Mistral—each with multiple incremental releases.

On top of that, we create sub-blocks also dependent on the task type that we evaluate the models on. The experiment includes three popular task types, namely Question Answering, Text Generation and Text Summarization.

With that being said, the independent variables manipulated in the experiment are categorized as shown in table 5:

Main Factor	Model Version
Co-Factor	Input Size
Blocking Factors	LLM Candidate, Task Type

Table 5: Experiment Design Factors

4.4.2 Treatment Structure. The combination of task type and input size forms the treatment in this experiment. For each LLM version, we test:

- **3 task types** (Paraphrasing, Extractive QA, Extractive Summarization)
- **2 input sizes** (Short and Long)

All combinations of task type and input size are tested within each sub-block. This results in **6 treatments per LLM version**. Across the 10 versions of LLMs, this gives us:

$$6 \text{ treatments} \times 10 \text{ versions} = 60 \text{ total treatments.}$$

4.4.3 Repetitions. To ensure the reliability of the results and to calculate average energy efficiency and resource utilization, each treatment is repeated 30 times. Therefore, the total number of experimental runs is:

$$60 \text{ treatments} \times 30 \text{ trials per treatment} = 1800 \text{ total runs.}$$

4.4.4 Summary of Experimental Design. The experiment systematically evaluates the interaction between *task type* and *input size* for each LLM version leading to a **MF-MT** design. The use of a blocked design ensures that comparisons across versions of LLMs are controlled for variability in model architecture, allowing for a robust analysis of the impact of incremental LLM releases on energy efficiency and resource utilization.

4.5 Data Analysis

The data collected throughout the experiment will undergo a systematic analysis to evaluate *energy efficiency* and *performance* across different Large Language Models (LLMs). *Energy efficiency* is derived from energy consumption using the following formula:

$$\text{Energy Efficiency} = \frac{\text{Response Time}}{\text{Total Energy Consumption}}$$

This formula allows us to quantify how efficiently each model performs a given task relative to its energy consumption. The following steps outline our approach to analyzing the results:

(1) **Data Exploration**: We will begin by examining descriptive statistics such as mean, median, and standard deviation for *energy consumption* (in joules), *execution time* (in seconds), *GPU utilization* in percentages, *CPU utilization* also in percentages, as well as *memory usage (both RAM and VRAM)* expressed in megabytes and lastly, *performance scores* of the actual models. Visual tools such as box plots and histograms will be used to display the distribution of each metric, allowing us to quickly identify trends and outliers across different LLM versions.

(2) **Normality Tests**: Before conducting any inferential statistics, the normality of the data will be assessed using *Q-Q plots* and the *Shapiro-Wilk test*, with an alpha threshold of 0.05. This step ensures that the assumptions of normality are met before proceeding to parametric tests. In cases where data is not normally distributed, non-parametric alternatives will be used.

(3) **Energy Efficiency Comparison**: For each inference task type (text generation, summarization, and question answering), we will compare *energy efficiency* across LLM versions. *Paired t-tests* will be used to determine if the mean differences in energy efficiency are statistically significant. In cases where normality assumptions are violated, the *Wilcoxon Signed Rank Test* will be applied instead. The effect size will be measured using *Cohen's d* to understand the magnitude of any detected differences.

(4) **Performance-Efficiency Correlation**: To assess the relationship between *performance metrics* (accuracy, response time) and *energy efficiency*, we will calculate *Pearson's correlation coefficient* if the data is normally distributed. If not, *Spearman's rank correlation* will be employed. This will help determine whether performance improvements lead to increased or decreased energy consumption across LLM versions.

(5) **Resource Utilization Analysis**: In addition to energy efficiency, we will analyze the *resource utilization metrics* (CPU and GPU usage, memory consumption). Paired t-tests or their non-parametric equivalents will be used to identify significant differences in resource usage between LLM versions, with the aim of understanding how these factors contribute to overall energy consumption.

(6) **Statistical Significance Testing**: For each analysis, we will use a significance level of 0.05. Any significant results will be further analyzed using post-hoc tests to pinpoint where differences lie among LLM versions. Additionally, effect sizes will be calculated using *Cohen's d* or *Cliff's delta* (for non-parametric tests) to provide insight into the practical significance of the results.

By combining these analyses, we aim to understand the trade-offs between *energy efficiency*, resource utilization, and model performance across successive LLM versions. The findings will be crucial in determining whether newer versions of LLMs achieve better energy efficiency without compromising performance.

5 EXPERIMENT EXECUTION

This section outlines the execution phase of the experiment. It details the procedures, tools, and methodologies employed

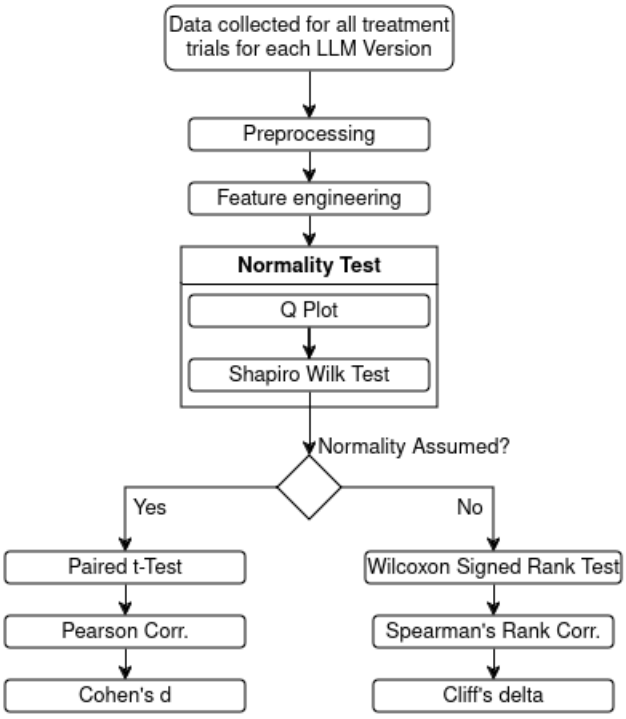

Figure 2: Data Analysis Process for LLM Energy Efficiency

to assess the energy efficiency and resource utilization of various LLMs.

5.1 Preparation

For our preparation phase, we installed the required libraries in order to run the models locally, using the same methodology as to reduce bias. The respective libraries enabled us to deploy the models using the hugging face transformers.

On the machine side, we had to configure the *top* tool in order for CPU metrics to show up as an averaged CPU utilization over all cores instead of single core usages.

5.2 Setup

To ensure accurate measurements, we minimized external resource consumption by terminating non-essential processes on the machine before running the experiments. This was critical in isolating the performance metrics of the LLMs under evaluation, allowing for a clearer understanding of each model's efficiency and resource demands.

With that being said, prior to launching our experiment we killed all background processes that were not kernel specific, using *top* as our tool to get an overview on said processes, and *kill* to remove the respective process.

Our hardware will have the following specifications:

- CPU: Intel(R) i9-13900KF
- RAM: 64 GB
- GPU: NVidia(R) GeForce RTX 4070
- VRAM: 12 GB

The tools used for this experiment are as follows:

- Experiment Runner: is a tool designed to automate the process of running the experiments, which lowers the possibility for bias or errors to appear during the execution of our experiments. By managing task scheduling and execution, this tool ensures that each model is evaluated under consistent conditions, facilitating a fair performance comparison across many versions.
- HuggingFace Evaluation: serves as our performance evaluation library that measures model outputs against predefined metrics. It provided a framework to assess how well each model performs in specific tasks such as text generation, summarization, and question answering.
- PowerJoular: used to extract both CPU as well as GPU power consumption. The metrics extracted from the tool are measured in Joules and it offers the energy consumption with a polling frequency of one second.
- Top: used to extract CPU and memory (RAM) utilization of the LLM model at process level, by taking the process ID of the LLM. This tool was configured to also have a polling frequency of one second.
- Nvidia-smi: used in order to extract GPU and VRAM utilization of the model during the process of inference. This tool was also configured to have a polling frequency of one second.

It is necessary to consider the data extraction frequency to accurately compute total consumption and utilization from our monitoring tools.

Below is a figure of the experimental components and how they communicate:

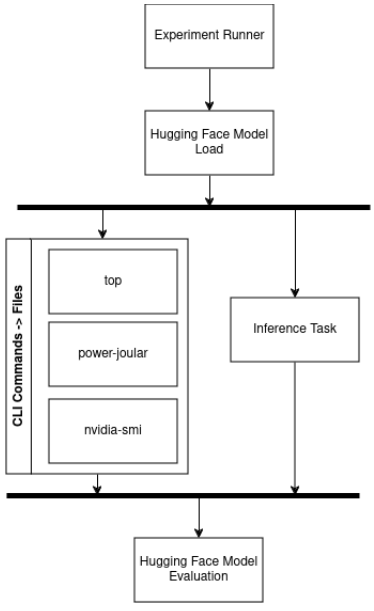

Figure 3: Experiment Execution Diagram

Experiment runner is setup to have the following sequence per each run:

- Load the model of interest (for example, Gemma v1);

- Start the metric extraction tools (namely top, powerjoular and nvidia-smi) and set them to pipe their output into files;
- Run the inference task with either the long or short input for the run's designated type of task (QA, Summarization or Text Generation);
- When the inference task is finished, turn off the metrics extraction tools;
- Call the huggingFace evaluation functions for the model and lastly, populate the data entry for the run.
- Wait for a cool-off period of 60 seconds before the next run.

5.3 Replication Package

Our Github repository[2] contains the data collected from the experiment as well as the script used to conduct the experiment.

You can also find all the necessary steps to reproduce our experiment in there.

6 RESULTS

In this section, we analyze the performance and energy efficiency of multiple large language model (LLM) versions across three primary tasks: text generation, question answering, and summarization, each tested with both short and long input sizes.

6.1 Energy Consumption Across LLM Versions

Energy consumption varied across model versions based on architecture and input size. Models such as *gemma-v1* and *qwen-v2* consumed significantly more energy for summarization tasks compared to generation or question-answering tasks (see Figure 4). Notably, energy requirements increased with larger input sizes, which underlines the computational load associated with processing extended text.

6.2 Model Performance Metrics

The GPU utilization, as depicted in Figure 5, remained steady across models with noticeable spikes for longer input sizes. Models such as *qwen-v2.5* recorded higher GPU utilization, particularly during summarization tasks. Similarly, CPU utilization remained consistently below 0.06% across all models and tasks, suggesting most of the processing load was handled by the GPU (see Figure 6).

6.3 Resource Utilization

Examining GPU and CPU power, Figure 7 reveals that models like *gemma-v1.1* and *mistral-v0.1* have notably high GPU power demands, particularly in generation tasks with long inputs. Additionally, VRAM usage, as shown in Figure 9, was markedly higher for long input sizes, with models such as *gemma-v2* and *qwen-v1.5* requiring substantial VRAM resources.

6.4 Response Time

Response time varied considerably across models and tasks. The *mistral-v0.1* model demonstrated the highest response times, especially in summarization tasks, as shown in Figure 10. This highlights the model's latency in handling extended summaries, which may impact real-time applications.

6.5 Performance and Energy Use Correlation

A strong correlation between GPU power consumption and total energy use is evident across different tasks, as seen in Figure 11. This suggests that managing GPU power could directly influence overall energy efficiency.

[2] https://github.com/andrei-calin-dragomir/greenlab-course-project.git

FIGURES

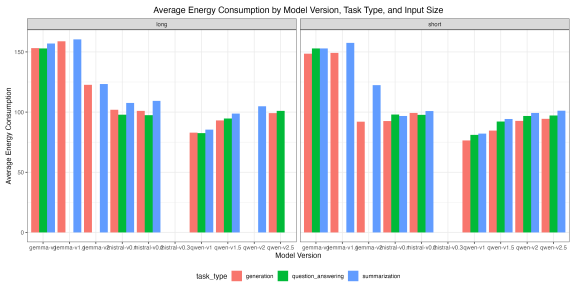

Figure 4: Average energy consumption by model version, task type, and input size.

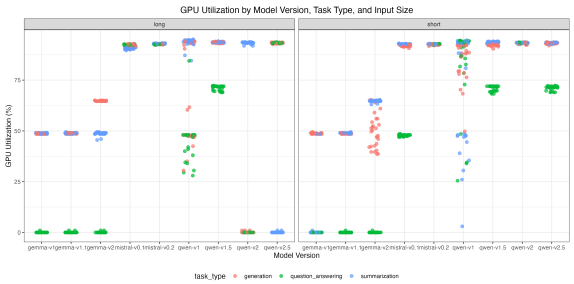

Figure 5: GPU utilization by model version, task type, and input size.

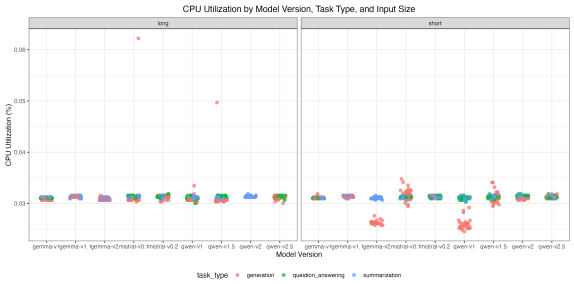

Figure 6: CPU utilization by model version, task type, and input size.

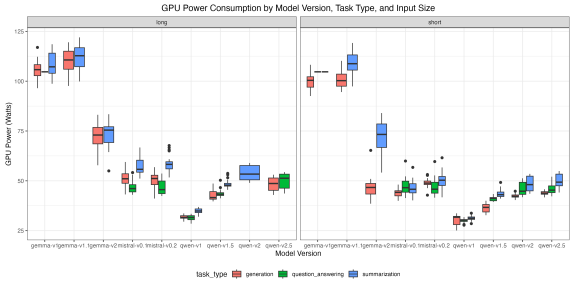

Figure 7: GPU power consumption by model version, task type, and input size.

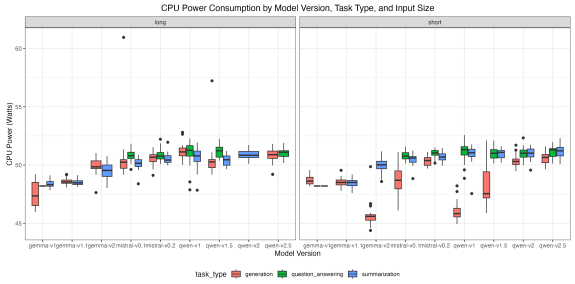

Figure 8: CPU power consumption by model version, task type, and input size.

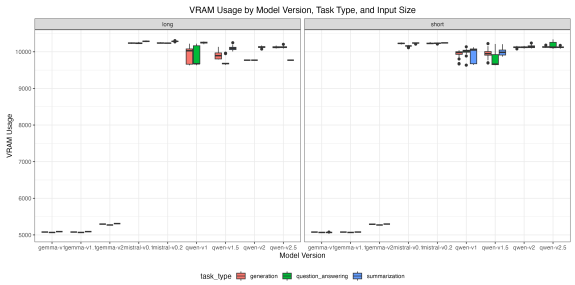

Figure 9: VRAM usage by model version, task type, and input size.

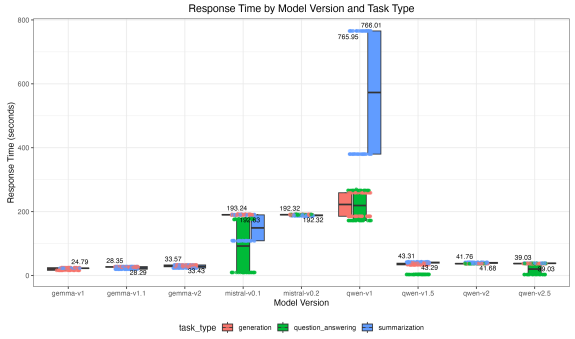

Figure 10: Response time by model version and task type.

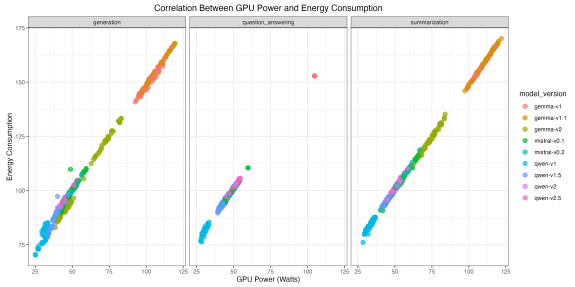

Figure 11: Correlation between GPU power consumption and energy consumption.

7 DISCUSSION

7.1 Key Findings Summary

Key results demonstrate a correlation between model version, input size, and hardware utilization metrics, especially GPU and CPU power and utilization. The energy consumption metric reveals a significant link with model complexity, suggesting that larger models with higher input sizes generally drive up power requirements across both CPU and GPU, highlighting the computational cost of larger, more complex models.

7.2 Interpretation of Results

The relationship between model parameters and energy consumption indicates that higher GPU utilization and VRAM usage are associated with increased energy demands. This suggests that GPU optimization is a critical factor in controlling power usage, especially for more complex model architectures. CPU utilization, though generally lower than GPU, also contributes to overall energy demands, which vary with model version and input size. The model versions with optimized architectures tend to show more efficient energy usage per unit of computational power, suggesting that hardware and model optimization can jointly contribute to sustainable computational practices.

7.3 Practical Implications

These findings underscore the importance of choosing optimal model versions and configurations for computational efficiency, especially when scaling models for production environments where energy consumption can impact operational costs. For applications requiring low energy usage, selecting models with moderate input sizes or more efficient architectures could result in considerable savings. Additionally, for organizations concerned with environmental sustainability, these insights suggest strategies for reducing the carbon footprint of model training and deployment through careful selection of hardware configurations and model versions.

7.4 Limitations

This study's limitations include a constrained range of model versions and input sizes, which may limit the generalizability of results to other models and architectures. The dataset also reflects specific hardware configurations, meaning the findings might not directly apply to other systems or configurations with different GPU or CPU capabilities. Additionally, the static nature of the dataset prevents real-time analysis of performance fluctuations under different operational conditions, such as variable workloads or power-saving settings, which may influence CPU and GPU utilization rates.

7.5 Future Work

Future research could expand this analysis by incorporating additional model versions and a wider range of input sizes to increase generalizability. Another area for exploration is the impact of real-time workload variation on hardware utilization and energy consumption, potentially leading to more adaptive and dynamic power management strategies. Investigating alternative hardware configurations, such as low-power GPUs or specialized AI accelerators, could provide further insights into minimizing energy consumption for deep learning applications.

7.6 Concluding Remarks

In conclusion, the study provides valuable insights into the relationship between model complexity, hardware utilization, and energy consumption. By selecting appropriate model versions and configurations, organizations can achieve a balance between computational efficiency and energy cost. The findings advocate for a more deliberate approach to model selection, especially for applications where energy usage and operational costs are paramount, aligning computational practices with sustainable and cost-effective operational strategies.

8 THREATS TO VALIDITY

The following section discusses various threats that might affect the validity of this research. For this discussion we will follow the classifications and described by Cook and Campbell[23].

8.1 Internal Validity

During our experimental process, at inference time, the metric extraction tools we used had a polling rate that was too high and sometimes unstable, rendering our data for the respective run to be too scarce. The lack of timestamps on the data points also creates a threat to the validity of our study due to the impossibility of creating an AUC[3] evaluation for the real consumption of the models. This applies for both GPU and VRAM utilization as well as power consumption and CPU utilization.

On the other hand, our input prompts were not selected from already existing benchmarks. Therefore, any variations in prompt complexity could affect metrics like response time or energy usage, making it harder to attribute these effects solely to the LLM version rather than differences in prompt characteristics.

We did not turn off CPU or GPU hardware configurations such as thermal throttling or hyper-threading and we did not collect data about the thermal status of the machine during the inference tasks. These aspects also have the potential to introduce bias in our readings, given that processors lose in energy efficiency[24].

8.2 External Validity

In terms of external validity, the generalizability of the study's findings is limited by the selected models and controlled setup. While Qwen, Gemma, and Mistral are evaluated, conclusions may not apply to other LLM architectures or versions, especially given the unique design and optimization approaches across models.

While the study focuses on common NLP tasks such as text generation, question answering, and summarization, these do not fully represent the variety of inference tasks in NLP. Tasks like dialogue modeling, sentiment analysis, and

[3] AUC = Area under the curve

language translation may impose distinct computational demands that differ from the selected tasks. Consequently, improvements observed in this experiment may not directly translate to other types, limiting the generalizability and construct validity of the findings. Thus, while the chosen tasks offer useful insights, they may only partially reflect the model's real-world capabilities across diverse applications where task requirements vary widely.

Further, the study uses instruct versions for Gemma, which are fine-tuned for specific task orientations. This specialization can alter the model's performance and resource efficiency, impacting external validity by potentially inflating its suitability in limited-use cases while limiting the generalization to broader applications.

On top of that, real-world deployments typically involve diverse hardware setups and usage scenarios, which are not fully reflected in this experimental environment. As a result, the energy efficiency observed here may differ under typical operational conditions, such as on cloud servers.

8.3 Construct Validity

The experiment emphasized GPU-based execution, as the LLMs were designed primarily for GPU deployment, making CPU metrics largely irrelevant. Consequently, the testbed's GPU was fully utilized across all runs, leading to saturation that obscured differences in resource demands between tasks and models. This ceiling effect in GPU utilization reduces the interpretive value of GPU metrics, as continuous high usage prevents detecting finer distinctions in GPU efficiency.

While a token limit was set, it was not enforced uniformly, resulting in variable output lengths across runs. Such variations can impact performance metrics like energy consumption and response time, as longer responses typically require greater computational resources. This inconsistency introduces bias, potentially masking genuine differences between model versions or tasks.

9 CONCLUSIONS

Our report reveals that energy consumption among large language models (LLMs) varies significantly depending on both the model version and the task type. Models such as gemma-v1 and qwen-v2 consume notably more energy during summarization tasks compared to text generation and question answering, suggesting that summarization is particularly resource-intensive. Similarly, GPU utilization is highest for models like qwen-v2.5, especially for summarization tasks, indicating that they require more computational power for complex text processing tasks. CPU utilization remains minimal across all models, confirming that GPU primarily handles the computational load. Additionally, certain models, including gemma-v1.1 and mistral-v0.1, demonstrate high GPU power usage in generation tasks with long input sizes, showing that power demands are linked to both task complexity and input length.

VRAM consumption also varies with models like gemma-v2 and qwen-v1.5 requiring more memory when processing larger inputs, emphasizing the importance of efficient memory management for performance optimization in high-resource tasks. Furthermore, a positive correlation between GPU power and energy consumption aligns with the expectation that higher GPU power demands lead to increased energy use, particularly in summarization tasks. Lastly, models like mistral-v0.1 show a slower response time for summarization tasks, indicating latency challenges with handling more complex inputs. Overall, these findings highlight the importance of selecting optimized models and configurations to balance energy efficiency and performance in different inference tasks.

REFERENCES

[1] T. B. Brown *et al.*, "Language models are few-shot learners," *arXiv preprint arXiv:2005.14165*, 2020. [Online]. Available: https://arxiv.org/abs/2005.14165

[2] A. Vaswani *et al.*, "Attention is all you need," *arXiv preprint arXiv:1706.03762*, 2017. [Online]. Available: https://arxiv.org/abs/1706.03762

[3] F. Duarte, "Number of chatgpt users (oct 2024)," *Exploding Topics*, 2024. [Online]. Available: https://explodingtopics.com/blog/chatgpt-users

[4] S. Uspenskyi, "Large language model statistics and numbers (2024)," *Springs*, 2024. [Online]. Available: https://springsapps.com/knowledge/large-language-model-statistics-and-numbers-2024

[5] G. V. Research, "Natural language processing market size, share & trends analysis report," *Grand View Research*, 2023. [Online]. Available: https://www.grandviewresearch.com/industry-analysis/natural-language-processing-nlp-market

[6] H. Touvron *et al.*, "Llama: Open and efficient foundation language models," *arXiv preprint arXiv:2302.13971*, 2023. [Online]. Available: https://arxiv.org/abs/2302.13971

[7] D. Patterson *et al.*, "The carbon footprint of machine learning training will plateau, then shrink," *arXiv preprint arXiv:2104.10350*, 2021. [Online]. Available: https://arxiv.org/abs/2104.10350

[8] A. de Vries, "The growing energy footprint of artificial intelligence," *Joule*, vol. 7, no. 10, pp. 2191–2194, 2023.

[9] X. Hu, P. Li, and Y. Sun, "Minimizing energy cost for green data center by exploring heterogeneous energy resource," *Journal of Modern Power Systems and Clean Energy*, vol. 9, no. 1, pp. 148–159, 2021.

[10] Y. Yuan, J. Zhang, Z. Zhang, K. Chen, J. Shi, V. Stoico, and I. Malavolta, "The impact of knowledge distillation on the energy consumption and runtime efficiency of nlp models," in *2024 IEEE/ACM 3rd International Conference on AI Engineering – Software Engineering for AI (CAIN)*, 2024, pp. 129–133.

[11] E. Strubell, A. Ganesh, and A. McCallum, "Energy and policy considerations for deep learning in nlp," *arXiv preprint arXiv:1906.02243*, 2019. [Online]. Available: https://arxiv.org/abs/1906.02243

[12] S. Han, H. Mao, and W. J. Dally, "Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding," *arXiv preprint arXiv:1510.00149*, 2016. [Online]. Available: https://arxiv.org/abs/1510.00149

[13] A. Lacoste, A. Luccioni, V. Schmidt, and T. Dandres, "Quantifying the carbon emissions of machine learning," *arXiv preprint arXiv:1910.09700*, 2019. [Online]. Available: https://arxiv.org/abs/1910.09700

[14] M. F. Argerich and M. Patiño-Martínez, "Measuring and improving the energy efficiency of large language models inference," *IEEE Access*, vol. 12, pp. 80 194–80 207, 2024.

[15] H.-Y. Lin and J. Voas, "Lower energy large language models (llms)," *Computer*, vol. 56, no. 10, pp. 14–16, 2023.

[16] S. Samsi, D. Zhao, J. McDonald, B. Li, A. Michaleas, M. Jones, W. Bergeron, J. Kepner, D. Tiwari, and V. Gadepally, "From words to watts: Benchmarking the energy costs of large language model inference," in *2023 IEEE High Performance Extreme Computing Conference (HPEC)*, 2023, pp. 1–9.

[17] Y. Xu, S. Martínez-Fernández, M. Martinez, and X. Franch, "Energy efficiency of training neural network architectures: An empirical study," *arXiv preprint arXiv:2302.00967*, 2023. [Online]. Available: https://arxiv.org/abs/2302.00967

[18] V. R. B. G. Caldiera and H. D. Rombach, "The goal question metric approach," *Encyclopedia of software engineering*, pp. 528–532, 1994.

[19] T. Hu and X.-H. Zhou, "5. unveiling llm evaluation focused on metrics: Challenges and solutions," *arXiv.org*, 2024.

[20] A. C. Community, "Alibaba cloud's qwen models attract over 90,000 enterprise adoptions within its first year," *Alibaba Cloud*, 2024. [Online]. Available: https://www.alibabacloud.com/blog/alibaba-clouds-qwen-models-attract-over-90000-enterprise-adoptions-within-its-first-year_601130

[21] Restack.io, "Mistral ai applications in industry," 2024, accessed: 2024-10-24. [Online]. Available: https://www.restack.io/p/artificial-intelligence-applications-answer-mistral-ai-industry
[22] A3logics, "The gemma model to democratize ai with open-source llms," *A3logics*, 2024. [Online]. Available: https://www.a3logics.com/blog/google-unveils-gemma/
[23] D. T. Campbell and T. D. Cook, "Quasi-experimentation," *Chicago, IL: Rand Mc-Nally*, vol. 1, no. 1, pp. 1–384, 1979.
[24] J. M. Cebrian and L. Natvig, "Temperature effects on on-chip energy measurements." IEEE, 2013, pp. 1–6.