Zaniar Ghaderi

Energy and Cost Estimation of Local Large Language Models

An Empirical Study for Business Decision-Making

School of Technology and Innovation

Master's thesis in Industrial System Analytics

Vaasa 2026

UNIVERSITY OF VAASA
School of Technology
Author: Zaniar Ghaderi
Title of the Thesis: Energy and Cost Estimation of Local Large Language Models : An Empirical Study for Business Decision-Making
Degree: Master's Degree
Programme: Industrial System Analytics
Supervisor: Petri Välisuo, Pradeep Bonda
Year: 2026 **Sivumäärä:** 71

ABSTRACT:

The rapid adoption of large language models in business organizations has created a growing need to understand the costs of running these tools locally or on cloud. While Online services of LLMs are widely available, organizations prefer to deploy the open-source models on their own hardware to protect data privacy and reduce risk of price changing. However, there is no practical framework to help businesses calculate their electricity costs of running LLMs locally.

This study measures the energy consumption of 18 locally deployed LLMs across three common business task types: text summarization, code generation, and financial analysis. The experiment was conducted using WSTAR datacenter in Vaasa, Finland, using 2 L40S NVIDIA GPUs. Energy consumption was recorded in real time using CodeCarbon library in Python and the results were translated into practical cost estimates for organizations of different sizes.

Finding shows that enabling chain-of-thought reasoning increases the number of output tokens by three to five times and raises total energy use, while energy per token remains constant. The monthly electricity cost for inference via local LLMs ranges from one euro for micro-sized companies to about 300 euros for enterprise-level organizations, based on Finnish energy price.

When comparing local deployment costs to online API services on a cost-per-million-token basis, online services were more cost-effective for almost all tested models. Through the analysis it is clear that energy cost savings are of not great importance while considering local vs cloud deployment decision, the more likely factors to consider are privacy and price risk mitigation.

KEYWORDS: Large Language Models, Energy Consumption, On-Premises Deployment, LLM benchmarking, Business Decision-making

Contents

Figures

Tables

Abbreviations

AI	Artificial Intelligence
LLM	Large Language Model
NLP	Natural Language Processing
GPT	Generative Pre-trained Transformer
API	Application Programming Interface
CoT	Chain-of-Thought
MoE	Mixture of Experts
GPU	Graphics Processing Unit
CPU	Central Processing Unit
RAM	Random Access Memory
VRAM	Video Random Access Memory
kWh	Kilowatt-hour
PUE	Power Usage Effectiveness
WUE	Water Usage Effectiveness
CO_2	Carbon Dioxide
NVML	NVIDIA Management Library
RAPL	Running Average Power Limit
IPMI	Intelligent Platform Management Interface
vLLM	Virtual Large Language Model serving framework
OpEx	Operational Expenditure

CapEx	Capital Expenditure
TCO	Total Cost of Ownership
SaaS	Software as a Service
FinQA	Financial Question Answering
SWE	Software Engineering
EPT	Energy Per Token
TPS	Tokens Per Second
KPI	Key Performance Indicator
DDR	Double Data Rate
WSTAR	Wasa Zero Emission Data Centre

1 Introduction

1.1 Context of the study

The introduction of large language models has been considered as one of the most significant technologies in the last decade. Since the release of GPT-3 by OpenAI in 2020, the adoption of AI-powered tools has grown. It attracted one million users within first five days of launch, 93% faster than any other technology (Curran et al., 2024). This rapid adoption was not only between public people. A systematic review of over 2300 academic publications between 2022 and 2024 has shown that LLMs have affected research across healthcare, education, and information technology in more than 80 countries. This review identified computational costs as one of the key challenges of this new product (Khan et al., 2024).

At the same time, businesses in many industries started using generative AI tools as part of their daily workflows. A survey reported by Alexander Bick et al. (2025) showed that nearly one third of the employees use generative AI for more than 60 minutes per day (Alexander Bick et al., 2025). As a result, understanding the costs of using these tools has become increasingly important for organizations to decide if it's beneficial to deploy these models locally.

A direct consequence of this widespread adoption is a significant rise in global energy consumption. Micrsofts's 2025 Environmental Sustainability Report documented a 23.4% increase in carbon emissions since its 2020 baseline which the company linked it mainly the growth of AI services and cloud infrastructure. Training and running large language models requires a large amount of energy that puts a heavy pressure on the datacenters across the world(Zachary Skidmore, 2026).

Because of these concerns, many organizations started looking for alternatives to cloud-based AI services. Running open-source large language models on local hardware has become an attractive option, because it ensures data privacy, regulatory compliance and better control over long-term costs. Availability of open-source models such as LLaMA, mistral, DeepSeek and Gemma has made local deployments more accessible even for small businesses who do not have access to large computational resources(Agrawal, 2025a; Manchanda et al., 2025). Software tools like llama.cpp and its platform Ollama have made it easier to run these models without any technical difficulties for users who are not AI specialists.

However, even though local deployment of large language models is becoming more attractive, businesses still do not have reliable and practical methods to estimate the electricity costs of running these models in real working conditions. Many existing research focuses on measuring energy consumption in large server environments or cloud-based systems, and task types, model settings, and organizational context were ignored or paid little attention to the them (Dauner & Socher, 2025; Husom et al., 2025; Ji & Jiang, 2026). This means that organizations that are considering local LLM deployment often cannot make good decisions on the cost of running them.

This thesis addresses this gap by providing an empirical study of the energy consumption of locally deployed large language models, with a focus on business-relevant tasks and practical cost estimation. The research was conducted at WSTAR zero-emission data center in Vaasa, Finland and its findings intended to help organizations of different sizes make better and more sustainable decisions using AI deployment.

1.2 Problem statement and objectives

The rapid adoption of large language models in business enterprises has created a growing need for organizations to measure and evaluate the operational costs of deploying LLMs locally. Online LLM API service providers offer a reasonable cost with scalable features, but they raise concerns related to data privacy, recurring subscription costs and product performance. As a result, many organizations are exploring the possibility of running open-source models on their own local hardware. However, there is no practical framework to measure or calculate the electricity costs associated with local LLMs.

Existing research on LLM energy consumption has largely focused on technical metrics measured in controlled server environments, with little attention given to translating their measurements into actionable cost estimates for organizations and businesses. Furthermore, most studies do not consider the variety of daily business tasks, model configuration, or operational settings that affect energy consumption in practice.

This study aims to fill this gap by empirically measuring the energy consumption of locally deployed LLMs across multiple business-related task types and translating the results into practical cost-estimation frameworks. Therefore, the specific objectives of this research are as follows:

- To measure the real-time energy consumption of multiple sets of locally deployed open-source LLMs across three business task types: text summarization, Numerical analysis, and code generation
- To develop a business cost estimation framework that translates energy measurement into operational electricity costs for organizations of different sizes.
- Comparing the cost of locally deployed LLMs with online LLM API services on a cost-per-million-token basis.

1.3 Research questions and research gaps

Despite the growing literature on LLM energy consumption, several important gaps remain. First, most existing studies measure energy consumption in large-scale cloud or server environments using frameworks such as vLLM and TensorRT-LLM, which are not representative of the local deployments of LLMs in small and medium-sized companies use. Second, previous studies rarely translate their electricity measurements to operational costs of business and there is a lack of framework for organizations to calculate or estimate the real cost of local deployment.

To address these gaps, this research is guided by the following research questions:

RQ1: How much energy do locally deployed open-source large language models consume per token across different business task types?

RQ2: How do model size, thinking capability, and keep-alive settings affect the energy consumption of locally deployed LLMs?

RQ3: What is the estimated daily and monthly electricity cost of running locally deployed LLMs for organizations of different sizes?

RQ4: How do the energy costs of locally deployed LLMs compare to the pricing of online LLM API services on a cost-per-million-token basis?

1.4 Technical contributions

This research is making several contributions to the existing body of work on energy measurement of local deployment of large language models.

1. This study aims to create an organized framework to combine three different business-related tasks, text summarization, code generation and numerical analysis. This

framework will create a benchmark to evaluate real-world business tasks while existing research relied on general academic benchmarks.

2. This experiment will be one of the first studies to empirically measure and compare the energy consumption of locally deployed LLMs, using real-world business-related tasks while monitoring the inference energy consumption of the computing resources with CodeCarbon library and GPU infrastructure.
3. A technical cost estimation framework will be developed that guides different sized companies to empirically estimate the operational electricity costs of running locally deployed LLMs. This framework calculates the electricity bills of real-world AI usage pattern for business decision-making.
4. A token-based cost comparison methodology will be developed to allow a direct and meaningful comparison between locally deployed large language models and commercial online LLM APIs. This framework will consider the full cost structure of AI products by using an evidence-based cost multiplier, which is drawn from industry reports and takes into account all major cost categories beyond just inference.

1.5 Scope and limitations

This study focuses on two primary goals. The first goal is to measure the total electricity consumption of locally deployed open-source LLMs during inference. The second goal is to translate the measurements of the experiments into actionable and practical calculations and estimations for businesses and organizations. In both cases, the scope is limited to text generation tasks that represent common business task flows, including text summarization, financial analysis and code generation.

The energy measurement in this study covers the total electricity consumed across all the hardware components during the inference, including GPU, CPU, RAM, and VRAM, as recorded by the CodeCarbon library. This provides a view of energy footprint of local LLM deployment.

Several limitations should be considered when interpreting the findings of this study.

1. This research does not measure or compare model accuracy, output quality, or inference latency. The experiments are not intended to be a performance benchmark, and the results should not be used to evaluate the quality of model outputs.

2. Hardware utilization metrics such as CPU usage percentage, GPU load, and RAM occupancy were not recorded. The experiments only measures electricity consumption in kilowatt hours and do not analyze how hardware resources are distributed during inference.
3. The study is limited to text generation tasks. Multimodal tasks involving image, video and voice generation are outside the scope of this study. Because these tasks require different computational workloads.
4. Only open-source models that are publicly available through the Ollama platform were tested. For online LLM API service costs, only equivalent of the locally deployed models which are available through OpenRouter platform were used for comparison.
5. The business cost analysis presented in this study covers operational electricity costs only and does not include capital expenditure such as hardware purchase cost, cooling infrastructure, or system maintenance. Organizations should therefore consider these additional costs when making deployment decisions based on the findings of this research.

1.6 Structure of thesis

This research is an empirical study to evaluate the deployment of large language models on local hardware. In the first chapter the background and introduction of the study were conducted as well as the research questions and objectives and limitations of the study. This chapter helps us to know the environment and concepts of the research area and the gaps that have existed in that field which are going to be filled in this study. In the second chapter, the related previously published papers were reviewed. Literature review is divided into five sections discussing LLM market in industry contexts, open-source LLMs and local deployment, Business costs and deployment strategies, energy consumption and sustainability of AI and core measurement and benchmarking papers. This section will lead to detect research gaps and related methodology that has been used in this research. On third section, the hardware infrastructure, benchmarks and the models, and the methodology of deployment and measurement of energy consumption were detailed. The related analysis and procedure to translate the results of the experiments into actionable business insights is also explained in this chapter. Chapter four contains the results of the experiment and analysis and the core findings of the research. This chapter concludes and combines the findings of the study. In the final chapter the summary of findings, recommendations, future studies, and technical contributions is explained.

2 Literature review

2.1 Introduction

Launching of LLMs is considered as a revolutionary event in history of Artificial Intelligence technology and usage of it spread fast and wide. While large-scale usage of LLMs is becoming necessary for companies, their costs are significantly rising. In this chapter, a brief literature survey on topics of LLM deployment strategies and business cost analytics is presented. The methodology followed is to collect and summarize papers on LLM market and industry context and how developers' society and businesses use open source and on-premises deployment of LLMs. Then, papers on the subject of business cost analysis of LLMs were assessed. At final section, research regarding measurement of energy consumption of LLMs were analyzed. The output of this literature review includes methods and strategies around deployment of LLMs on local hardware and measuring the energy consumption of LLMs while running inference workloads.

2.2 LLM Market & Industry Context

In this section the question of how LLMs are introduced and spread and how they are being used in an industry context will be answered. The impact on energy usage and environmental effects will be reviewed as well.

Since introduction of GPT-3 by OpenAI in 2020, the usage has increased significantly and attracted 1 million users in just 5 days, 93% faster than its closest competitor, which is Instagram that took 75 days to gain 1 million users (Curran et al., 2024). Khan et al. (2024) reviewed 2377 articles from Scopus and Web of Science between the years 2022 and 2024 and analyzed 1161 of them to measure the impact of ChatGPT across research disciplines globally. Their findings show that ChatGPT affected 38.6% of the papers in healthcare, 18.6% of IT/Computer Science and 17.3% of education/research papers. Over 80 countries have contributed to their research leading by USA, China and India. The paper documents that LLM adoption is rising across industrial and operational sectors and noted that computational costs of running AI as a significant challenge (Khan et al., 2024).

Microsoft 2025 Environmental Sustainability Report shows the rise in energy consumption after launching large language models. Emissions rose 23.4% since its 2020 baseline and the company

considers this growth is related to AI and cloud expansion, while this company count this rising in energy usage, that is around 168%, as a modest increase (Zachary Skidmore, 2026). Overall, generating emissions and energy consumption shows a significant rise after launching LLM services.

2.3 Open-Source LLMs & Local Deployment

In this section, the importance of open-source LLMs and Local Deployment will be assessed. Some of the reasons for local deployment include data privacy, accessibility, and transparency, counted as the most important for deployment of local LLMs.

With wide spread adoption of large language models, demand for open source LLMs raised and this was due to customization of models for specific jobs, tasks and privacy, offline access and cost control are cited as other important reasons (Agrawal, 2025b; Yu et al., 2023). Closed-source models are often being used because of their proprietary safeguard, security and regulatory compliance. While open-source alternatives consider transparency, contribution of developers and accessibility as an important factor. Open-source LLMs not only enhance transparency and reproducibility but also helps to create a more accessible, collaborative and equitable research ecosystem (Manchanda et al., 2025, p. 9).

Agrawal et al. (2025) investigated the implementation of different models on local hardware and published a useful documentation of implementing multiple models on multiple hardware specifications, software tools and local vs cloud trade-off. They have deployed LLaMA-3.3 70B, Mistral 7B, and Qwen 2.5 as the most trending open-source models in 2024, in three hardware categories. Low-end category with 8-16 GB RAM without any GPUs (CPU only) for models with up to 7 billion parameters. Mid-range with 16-32 GB with mid-tier GPU for up to 30B models and high-end (64+ GB RAM with GPU model RTX 3090/4090) for 70B models. The paper concluded that using higher GPUs increases the energy usage and acknowledged this as a persistent challenge (Agrawal, 2025b).

Touvron et al introduced LLaMA, their open-source language model family, that ranges from 7B to 65B parameters. They trained these models using online and open access datasets with accessible hardware. Their core finding includes one of their models (LLaMA-13B) outperformed GPT-3 (175B) on most benchmarks despite being 10 times smaller. This finding proof that efficient training on more tokens can compensate for fewer parameters (Touvron et al., 2023, p. 11). They

also calculated the energy consumption and carbon footprint while training their models. Training LLaMA-7B consumed 36 MWh and emitted approximately 14 tCO_2eq, while LLaMA-65B consumed 449 MWh and emitted 173 tCO_2eq, both using A100-80GB GPUs drawing 400W each (Touvron et al., 2023, p. 10).

Open-source large language models are popular among the open-source community because of their transparency and data privacy. But deployment of this kind of model on local hardware costs a considerable level of energy and generates a huge amount of carbon emissions. For business a better alternative may be having open source LLM that are locally deployed that will negate issues with data handling, security, compliance and confidentiality. But for businesses to make this decision they need proper methodology to analyze the tradeoffs needs to be developed as standard methodology has been yet formalized.

2.4 Business Cost Analysis & Deployment Strategy

Organizations must choose between the comfort of using cloud-based AI and the long-term financial control of on-premises hardware.

Pan et al. (2025) created a cost-benefit framework to compare deployment of open-source and on-premises large language models with API-based LLM use. They calculated power usage for 9 open-source LLMs including different models from Llama, Qwen, and Mistral and compare it to cloud APIs like GPT-5, Claude and Gemini. They Also calculated the cost of hardware they were using and an estimation of the electricity usage for a simulated workplace. The input and output tokens were counted and compared to online LLM API's tariffs, and the break-even points were analyzed for 54 combinations. Their findings show that performance gap between small and large open-source models is much smaller than their hardware cost difference. For instance, a 30B model often gets more than 90% of capabilities of a 70B model in a smaller cost difference (Pan et al., 2025). This research assumes that the cost of electricity is constant and predictable and calculated it by multiplying GPU wattage by hours. Different models consume different amounts of energy. The simulated workload does to contain business specific tasks which are common like text summarization and financial analysis.

Zhang et al. (2025) in their research developed a mathematical model to study strategic business decisions faced by LLM providers. The main goal is to answer the questions of whether they should offer only cloud-based LLMs or allow them to run local LLMs on their local hardware. The

effects of this decision on the provider's profit and user's benefit were analyzed by implementing a game theory model based on the simulation of providers and users and tested two scenarios. In one scenario there were only Cloud LLMs available, while in the other scenario, there was a possibility for users to run the open-source models on their own hardware. Users with different characteristics including privacy concerns, were added to the scenario. Their core finding was that offering local deployment reduces the provider's profit in most cases, because potential users will move away from the paid cloud services. However, it increases overall user benefits because users gain more freedom and face less limitations that the cloud LLMs (Z. (Jack) Zhang et al., 2025). One of the limitations of this work is it just evaluated the theoretical benefit of the local deployment of large language models and did not mention any electricity usage or business approaches. They did not factor in costs for local deployment of LLM.

Researchers at Lenovo in a technical whitepaper (Sachin Gopal Wani et al., 2025) called "on-premises vs cloud: Generative AI total cost of ownership" compared the running generative AI models on local hardware against using cloud services from providers such as AWS, Google cloud and Microsoft Azure. The goal was to publish a document to help organizations decide which deployment option is more financially practical. They selected 7 real and different server configurations with NVIDIA's GPUs such as H100, H200, and L40S and compared each of them to cloud instances. Then they analyzed three scenarios. First, they calculated break-even points and the exact time of on-premises LLM will be cheaper than cloud LLMs. Second, the total savings after 5 years considering 24-hour operation per day were calculated. Third, the minimum daily usage hours required for local deployment to be more economical than cloud services were calculated. Their findings show that, cloud services are better choices for short-term or irregular workloads, while on-premise deployments are more sustained in long-term AI operations (Sachin Gopal Wani et al., 2025). This study does not provide an empirical measurement of electricity consumption during local Large Language Model (LLM) inference; rather, it offers a theoretical estimation based on existing hardware benchmarks.

Enterprise strategy group, commissioned by Dell Technologies, in their report (2025), compared local running of LLMs on Dell's local hardware against two online cloud API providers, Amazon AWS and OpenAI. They deployed an open-source model, Llama 3 with 70B parameters, on their local hardware and operate it as an AI-powered chatbot for three organization sizes: Low-usage (5,000 users), medium usage (10,000 users), and 50,000 users (high usage). They assumed the

usage of each user will be 50 queries per day with an average of 3,000 tokens per query. The paper analyzed all costs over a 4-year period and counted all the costs including hardware purchase, power and cooling costs (at $0.162 per kilowatt-hour as a standard), software licenses, system administration, and cloud instances fees. The core finding of this practice shows that local deployments is 3 times more cost-effective than using cloud services and the financial advantages of local deployments grows significantly as the number of users increases (Kaufmann, 2025). The limitation of this study contains that it does not provide a technical measuring of the electricity consumption of the on-premises LLMs, while it offers an estimation of the token usage from the users of a chatbot in an organization.

In conclusion the data suggests that for a regular workload, on-premises models can be three times cheaper than cloud APIs over several years.

2.5 Energy Consumption & Sustainability of AI

Measuring the environmental footprint of AI requires looking beyond model size to include infrastructure overhead and real-time hardware metrics. While larger models generally use more power, factors like "reasoning mode" and token output volume are the primary features of energy consumption and carbon emissions.

Jegham et al. conducted research in 2025 calculating Power Usage Effectiveness (PUE), Water Usage Effectiveness (WUE) and carbon intensity of an AI query to estimate the energy usage of a single prompt. They first combined the timing data of 30 major LLM models using 3 different (short, medium, long) prompt lengths. The collected latency time (how long before the first word appears) and how many words per second are being generated from a public benchmarking website (Artificial Analysis). Combining the response time and the power draw of each hardware that a company is likely to use, gives energy consumption in watt-hour (Wh). This simulation has been run 10,000 times using slightly varied inputs to consider the real-world unpredictability. Then the infrastructure energy consumption including cooling system, lighting, and power distribution added to the measurements. This measurement applied with each company’s published efficiency ratios (PUE, WUE, CIF). Their core findings show that infrastructure matters as much as model sizes and reasoning models are more expensive to run than nonreasoning models. For instance, GPT-5 on “high reasoning” mode uses 7 times more energy than on “minimal” mode for the same query (Jegham et al., 2025). One of the limitations this study contains is that it is

not measuring real time energy consumption on local hardware. The researchers estimate the electricity consumption by multiplying time needed to answer the prompt and the hardware electricity wattage.

Ji and Jiang (2026) in their systematic review paper, collected useful information on electricity usage of LLMs during their lifecycle, from training to inference. They also identified key challenges in measuring and supplying electricity and proposed solutions to make LLM power use more sustainable. Two main ways to measure energy consumption were 1. Online Measurement (Direct) which includes measuring the electricity in real time while the model is running from the hardware, and 2. Offline Estimation (Formula-based) which will be calculated by multiplying the time of running and power draw of the hardware. They have found that inference time is using 60% to 90% of total lifecycle energy, while the rest of it is for training the model. The research concluded that bigger models consume more electricity, but the relationship between the number of parameters and the energy consumption is not perfectly linear (Ji & Jiang, 2026). This paper calls for LLM developers, data center operators, and policymakers to work together to establish common reporting standards because there is no standard method for measuring LLM energy consumption.

In research by Alexander Bick et al. a survey was conducted to find out what percentage of people in the workplaces use generative AI to do their tasks. In this real-time population survey, researchers asked employees how much time they spent using AI at work. Among people who used AI at least one time in the last month, 31.9% voted for more than one hour per day, 47% selected between 15 and 59 minutes per day and 21% selected under 14 minutes of daily AI usage at work (Alexander Bick et al., 2025).

The economic structure of AI products reported that costs of running an LLM are distributed across several operational categories. According to a survey of 202 AI product teams, costs of inference of LLMs makes approximately 23% of total AI product costs for both general availability and scaling stage (ICONIQ, 2025).

Dauner and Socher (2025) ran 14 different local LLMs including Qwen, Deepseek, Llama and Cogito LLM families on a local NVIDIA A100 GPU. They measured the CO_2 emissions produced by the LLMs during the inference. Their goal was to find a relationship between a model's accuracy, its size, and its environmental cost. Researchers gave each model 1,000 questions from the MMLU benchmark in five subjects. Each question was answered in two ways, first selecting from

multiple-choice options and second by writing a free form response. The evaluation for correctness of the free responses was done using OpenAI's o4-mini as an automated judge (Dauner & Socher, 2025). Their findings show that larger and reasoning-enabled models are more accurate but produce significantly more emissions, mainly because of generating more text tokens. The key conclusion is that the output volume token is the primary factor in predicting emissions. This study has multiple limitations, including no infrastructure overhead energy consumption was considered. Besides GPU energy usage, there are cooling systems, power distribution and lighting energy consumption which was ignored for this paper. In addition, models were limited to 7-72 billion parameters while there have been many larger models published during 2025. Using only text-based academic prompts and not analyzing business context or cost analysis from business approach is another limitation for this research.

Fadel Argerich and Patino-Martinez (2024) introduced their software tool named EnergyMeter to directly measure the energy consumption of LLMs during inference and used it to run systematic experiments to investigate how different technical factors affect energy consumption of different LLMs. EnergyMeter is a Python tool that reads data from multiple hardware components including CPU (via Intel RAPL), GPU (via NVIDIA's management library), memory (also via RAPL), and storage (via custom eBPF-based tracking method) to measure the real time energy consumption while running LLMs. The Hardware specification was an Intel i9 CPU and NVIDIA RTX 4090 GPU. They loaded open-source models including Pythia, Bloom, LLaMa2, Dolly, and RedPajama, by using HuggingFace as the running platform. They used simple 8 token prompts extracted from the Pile, an 825 GB English text dataset commonly used to train LLMs and created a simple question-answering application to measure the energy consumption of LLMs. They used published papers on arXiv in August 2023 and asked manually written questions from the papers (Argerich & Patiño-Martínez, 2024). The prompts and the questions they used are a challenge to validation of the benchmark. One major limitation of the study is that it does not consider tasks like text or email summarization.

In another study, Husom et al. (2025) deployed 28 quantified LLMs on a Raspberry Pi 4 and measured the real time energy consumption, accuracy, and response speed of multiple task types. They used three model families, LLaMA 3.2, Qwen 2.5, and Gemma 2, which range from 0.5 to 2.6 billion parameters. The researchers used multiple quantization levels, from full precision (FP16) down to aggressive 2-bit compression (q3). All models were run using Ollama framework

on an edge device with 4 GB of RAM. Energy was measured using Joulescope, a physical power meter device connected in series between Raspberry Pi and its power supply. Each model answered questions from five standard benchmark datasets. Their core findings include that quantization significantly reduces energy consumption and accuracy of doing the task was highly dependent on the task regardless of quantization level (Husom et al., 2025). They limited the experiment to the embedded devices like Raspberry Pi and the variety of the models was low. In addition, no cost analysis from business point of view was mentioned in the paper.

Energy efficiency in AI is largely determined by operational choices, such as quantization and prompt length, rather than just the number of parameters.

2.6 Core measurement and benchmarking papers

Many researchers are now studying how much electricity AI models use, looking at how different hardware, software, and model sizes affect energy costs and speed.

Samsi et al. in their research in 2023 tried to benchmark inference performance and energy costs of different sizes of LLaMA models on two generations of GPUs which were NVIDIA V100 and A100 using two datasets (Alpaca and GSM8K). They have used multi-node, multi-GPU deployment inference across up to 32 GPUs. Their core findings include increasing the number of GPU, always increasing energy consumption but in a linear fashion. For instance, running LLaMA with 65B parameters on 8 GPUs takes 300 watts of power, while running the same model on 32 GPUs takes 1 kilowatt of power. So, businesses should be aware that hardware scaling directly affects the electricity costs. In addition, businesses should consider a trade-off between speed and energy efficiency. Using A100 generation GPUs shows significant improvements in inference throughput over V100, but the energy cost per second will increase after this improvement. They also reduced the power usage by 30% from 250W to 175W and measured the time of inference. On average it increases by 6.7%. This is highly actionable for businesses that want to lower operating costs. A modest performance sacrifice leads to substantial energy savings (Samsi et al., 2023). Regarding the limitations of this study, the researchers used only LLaMA family models and focused on technical metrics rather than busiess metrics like costs.

Niu et al. (2025) introduced TokenPowerBench, a lightweight and open-source benchmark for measuring LLM inference power. Their goal is to provide a way for users to measure exactly how

many Joules they are spending per token. On Figure 1 you can check an overview of how TokenPowerBench works.

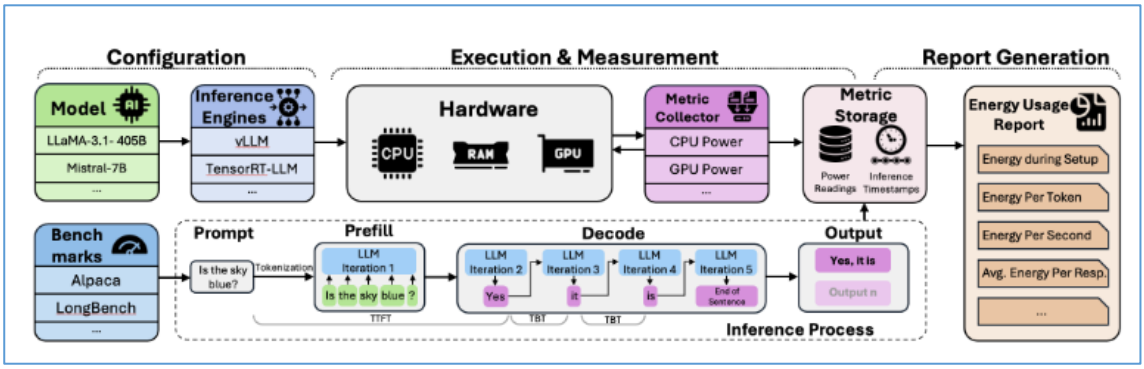

Figure 1 Overview of the TokenPowerBench Architecture (Niu, Zhang, Li, et al., 2025, p. 4)

This research tested the tool on multiple popular models, such as Llama, Falcon, Qwen, and Mistral, ranging from small versions to large ones like Llama 3-405B using Alpaca and LongBench as their datasets. Alpaca contains 52,002 short, chat-style prompts that are used to simulate typical interactive conversations and LongBench is an open-source benchmark featuring longer context, with prompts reaching up to 10,000 tokens. It is used to test how extended input affects energy consumption of LLMs. The tool supports four main software engine to run LLMs including vLLM (Kwon et al., 2023), TensorRT-LLM (Neal Vaidya et al., 2023), DeepSpeed (Aminabadi et al., 2022), and Transformers (Face H, 2024). The core findings of this study are that larger models are much more expensive than smaller ones. Increasing the size of the parameters from 1 billion to 70 billion parameters increases the energy consumption per token by 7.3x. In addition, asking multiple questions at the same time decreases the energy consumption because of sharing the cost across more tokens. Software choice can reduce energy consumption per token by 25-40% compared to standard models (Niu, Zhang, Li, et al., 2025). Among the limitations of this study, it should be noted that analyzing accuracy vs energy trade-off and measuring electricity consumption from business approaches were ignored in this study.

In another study, researchers conducted a large-scale measurement to find a reasonable relationship between different factors and energy consumption of LLMs. They tried 155 different model architecture with 21 GPU configurations. In total they measured energy use of 32,500 different combinations over nearly one million prompts (Caravaca et al., 2025). They developed a browser extension to help users understand the environmental impact of their AI interactions. This study deployed LLMs using vLLM framework on cloud-based GPU environments on Google

instances. They used Codecarbon library and NVIDIA management library in python to measure energy consumption. They tested multiple variables for each measurement including input and output, token lengths, batch size, different model architecture, various GPU types, and quantization methods. Their Machine learning models worked well on estimating the electricity usage. Random Forest achieved accuracy of 97% on train data and 92% on test data (Caravaca et al., 2025). Their core findings show that output tokens consume significantly more energy than input tokens (approximately 11 times more for long outputs). One of the limitations of this study is that they have not deployed LLMs on local hardware as well as not measuring the business side of the energy consumption and cost of ownership of the system.

Niu et al. (2025) did research to measure energy consumption across four most used LLM inference engines: vLLM, TensorRT-LLM, DeepSpeed, and Transformers. They measured two stages of the inference lifecycle, the setup stage (engine initialization and model loading) and the token generation stage. They tested three Llama models (1B, 3B, and 8B) on a system with NVIDIA H100 GPUs and measured energy consumption across GPU, CPU, and DRAM. They used the Alpaca dataset with 52,002 prompts and tested three workload scenarios: Standard Load (batch size 128, 500 tokens), High Concurrency (batch size 256, 500 tokens), and High Throughput (batch size 256, 2000 tokens) (Niu, Zhang, Zhao, et al., 2025). They used IPMI for total system power measurement, NVIDIA Management library for GPU power consumption, and Intel RAPL for CPU and DRAM measurement. Each experiment was repeated 100 times, and they measured setup and token generation phases of the inference across multiple metrics including latency, energy per token, energy per response, energy per second, and the energy-throughput ratio. This breakdown allows them to understand which hardware component consumed most energy at each stage of the inference process. Their core findings show that no single inference engine is universally best and it depends on the use case as well as GPU takes 50% of the energy consumption followed by CPU with 12-13% (Niu, Zhang, Zhao, et al., 2025). This research ignored the business context of the study, and it does not relate energy consumption to business use cases, ROI, or operational costs. It misses cost analysis like converting energy consumption into monetary costs or carbon emissions and does not compare local deployment vs using online APIs.

Pronk and Zhao (2026), developed the “LLM Efficiency Benchmark” to evaluate LLM energy consumption under realistic operating conditions using vLLM. Their main goal was to focus on three main factors: the number of concurrent requests, model size, and model architecture. Used

hardware contained 2 NVIDIA RTX 3090 GPUs using the HellaSwag dataset for inference tasks. They measured energy consumption using CodeCarbon by reading low-level sensors like NVIDIA-smi, and Intel Powergadget) every 15 second across GPU, CPU, and DRAM components. Each experiment contains 200 warm-up requests to reach thermal saturation before measurement. Then the queries were sent in different number of simultaneous requests, from 5 to 5000, to simulate real world situation. vLLM gets the requests and they measure energy consumption per request in joules. Key findings show that with concurrent load, energy consumption decreases significantly and model size has a close to linear relationship with energy consumption (Pronk & Zhao, 2025). The limitation of this study is that no larger models were used in experiments they limited model sizes to than 6.9B parameters and not mention business contexts like cost of ownerships, ROI, or any analysis on energy consumption vs business outcomes.

In a research study conducted by Fernandez et al. (2026), a comprehensive analysis of energy consumption for LLM inference through different situations of real-world workloads was studied. They focused on multiple components including software framework, decoding strategies, GPU architecture, model parallelization and model architecture to measure which factor has the most effect on energy efficiency. They tested models from 1B to 32B parameters using Llama and Qwen families, across classical NLP tasks and real-world LLM workloads including conversational AI and Code generation. They used CodeCarbon and NVIDIA Management library to measure GPU energy consumption, dividing it into two stages, prefill and token generation. Their core findings show optimizing energy consumption is highly dependent on workload and can be reduced by 73%. Output token generation is significantly more energy-intensive than prefill. Mixture-of-Experts (MOE) models consume 54.24% energy than using single models (Fernandez et al., 2025). This paper has multiple limitations including no cost analysis from business approaches like RoI or business value calculations and ignoring measuring total cost of ownership. The range of the used models limited to 32B parameters while more complex models has been released during the publication of this paper. No local deployment of LLMs can be considered as a major limitation compared to the current research.

The studies show that while larger AI models use more power, businesses can save a lot of energy and money by choosing the right software, processing many questions at once, or slightly reducing the speed of the hardware. In Table 1 there is a summarization of the papers that tested and evaluated deployment of LLMs and their considerable factors for the current research.

Table 1 summary of the papers that deployed LLMs and measured energy consumption.

Research Paper	LLM Engine	Benchmark dataset	Measurement tools	Models Used	Hardware Used
Pan et al. (2025)	Not specified	Simulated workplace queries	Formula-based estimation	Llama, Qwen, Mistral) vs. GPT-5, Claude, Gemini	Not specified (On-premises hardware)
Wani et al. (2025)	Not specified	Real-world business scenarios	Theoretical estimation based on hardware benchmarks	Generative AI models	NVIDIA H100, H200, L40S
Jegham et al. (2025)	Not specified	Short, medium, and long prompts	PUE/WUE/CIF formulas	30 major LLMs (including GPT-5)	General company hardware
Dauner & Socher (2025)	Not specified	MMLU	OpenAI o4-mini (as accuracy judge)	14 models (Qwen, Deepseek, Llama, Cogito)	NVIDIA A100 GPU
Argerich & Patiño-Martínez (2024)	HuggingFace Transformers	The Pile (825 GB English text dataset)	EnergyMeter	Pythia, Bloom, LLaMA2, Dolly, RedPajama	Intel i9 CPU, NVIDIA RTX 4090
Husom et al. (2025)	Ollama	5 Standard Benchmarks (e.g., GSM8K, Hellaswag)	Joulescope	LLaMA 3.2, Qwen 2.5, Gemma 2 (0.5B - 2.6B)	Raspberry Pi 4 (4GB RAM)
Samsi et al. (2023)	Multi-node/ Multi-GPU Sharding	Alpaca and GSM8K	Not specified	LLaMA (various sizes up to 65B)	NVIDIA V100 & A100 (up to 32 GPUs)
Niu et al. (2025a)	vLLM, TensorRT-LLM, DeepSpeed, Transformers	Alpaca and LongBench	TokenPowerBench	Llama, Falcon, Qwen, Mistral (up to 405B)	NVIDIA H100
Caravaca et al. (2025)	vLLM (Google Cloud Instances)	1 Million Prompts (various lengths)	CodeCarbon	155 architectures	21 GPU configurations (Google Cloud)
Niu et al. (2025b)	vLLM, TensorRT-LLM, DeepSpeed, Transformers	Alpaca	CodeCarbon	Not specified	NVIDIA H100
Pronk & Zhao (2026)	vLLM	HellaSwag	CodeCarbon, NVIDIA-smi, Intel Powergadget	Various (up to 6.9B parameters)	2x NVIDIA RTX 3090
Fernandez et al. (2026)	Not specified	Classical NLP tasks, Conversational AI, and Code Generation	CodeCarbon	Llama and Qwen families (1B to 32B)	Not specified (Multiple GPU arch.)

3 Methods

In this chapter the methodology of this research study and the experiment process are explained. The infrastructure and benchmarks used were explained in sections 1 and 2. The instructions for deploying LLMs on local hardware are explained in section 3 and the energy measurement tools were mentioned in section 4. Final chapter will explain the methodology of business cost analysis and the way results are applied to real business cases.

3.1 WSTAR environment

Wasa Zero Emission Data Center (WSTAR) is a research council of Finland and European union funded project that enables research in future climate-neutral data center. WSTAR models data centers with tight integration with the local energy systems. This project was funded by 1.9 million euros by European unions and started on January 2023, ended on December 2025 (*WSTAR – Wasa Zero Emission Data Centre*, 2025).

The WSTAR data center contains a dedicated GPU computing rack (Rack 3) designed for high-performance computing tasks. This rack houses the hardware used for running and testing local large language models. The rack includes the following components:

Rack Enclosure:

- HPE 42U 600x1200mm Advanced G2 Shock Rack (Model: P9K10A)
- Power distribution unit: HPE/EN G3 Metered 3-Phase 11kVA PDU (Model: P59412-B21)
- Network switch: Aruba 6300M 48G PoE+ (Model: JL762A)

Server Hardware

The rack contains two identical HPE ProLiant DL320 Gen11 GPU CTO Servers (Model: P61218-B21). Each server has the following specifications:

- **Processing Power:**
    - CPU: 1 × Intel Xeon-G 5412U Processor (Model: P49618-B21)
    - GPU: 2 × NVIDIA L40S 48GB PCIe Accelerators (Model: S2L70C)
- **Memory:**
    - RAM: 8 × 32GB = 256 GB DDR5-4800 Smart Kit (Model: P43328-B21)
    - Total memory per rack: 512 GB

- **Storage:**
    - Storage drives: 2 × 1.92TB NVMe RI SFF SSDs (Model: P64844-B21)
    - Total storage per server: 7.68TB
- **Additional Components:**
    - Storage Controller: HPE MR416i-p Gen11 SPDM Controllers (Model: P47777-B21)
    - Network Adapters: BCM 5719 1Gb 4-port BASE-T OCP Adapters (Model: P51181-B21)
    - Power Supply: 1800-2200W Titanium Hot-Plug PSUs (Model: P44712-B21)
    - Management System: HPE iLO Advanced
    - This GPU rack provides significant computing power for running local LLMs:
    - Total processors: 2 CPUs
    - Total GPU accelerators: 4 NVIDIA L40S cards (192 GB total GPU memory)
    - Total system memory: 512 GB DDR5
    - Total storage: approximately 15.36 TB NVMe SSD

This configuration allows researchers to deploy and test multiple large language models, while measuring their energy consumption under different workload conditions. Figure 2 shows a view of the WSTAR datacenter and its three racks. The middle rack contains 2 L40S NVIDIA GPUs.

Figure 2 A view of WSTAR

3.2 Benchmark dataset

In this research, three business tasks have been evaluated. Text summarization, numerical report analysis and code generation tasks. Each of these tasks considered to be major tasks that are being used in daily business tasks (Busch & Leopold, 2024).

3.2.1 Text summarization tasks

The first task was text summarization. Text summarization is known as one of the most important tasks in natural language processing. It is defined as the process of extracting the most important and meaningful information of a source (or sources) to deliver a shorter version of it to user (or users) (H. Zhang et al., 2025). Text summarization in business can be used for summarizing emails, documents, and reports. In this research, a text summarization dataset named WikiHow has been used (Koupaee & Wang, 2018). It contains 230,000 pairs id articles and summaries from Wiki-How.com website. Koupaee and Wang used a python web scraping tool to collect the data and tested the existing summarization methods on their own dataset comparing to other benchmark (e.g. CNN/Dail Mail dataset).

The format of the data is in a pair of articles and summaries. Therefore, to create a useful prompting each article was passed to the LLM in the following format:

```
You are a technical writer specializing in the WikiHow format. Your goal is to convert complex information into clear, chronological, and actionable steps. Use a helpful, instructional tone and ensure each step begins with a strong imperative verb.

Summarize this WikiHow article into a headline:

{text}

Summary:
```

Figure 3 shows a screenshot of the WikiHow website which is the resource of instructions for different purposes.

Figure 3 A screenshot of WikiHow website

3.2.2 Analysis of numerical report tasks

Analyzing business reports and numerical documents is another task that has been considered to simulate business-related queries on LLMs. FinQA, Question-answering pairs over Financial reports was the dataset that has been used for this task(Chen et al., 2022). Chen et al. collected 8,281 question-answer pairs from real financial reports of S&P 500 companies. Every question requires reading the text and understanding the tables, then performing mathematical calculations to find the answers. They first selected 12,719 report pages from the publicly available earning reports from S&P 500 companies (1999-2019) and hired 11 financial professionals through UpWork platform to design questions and answers. They also added a step-by-step calculation program for each question. They found the FinQA dataset very challenging because of a 30% gap between the accuracy of financial experts and best computer model (FinQANet with RoBERTa-large). The data contains multiple sections as JSON files. The following format to create a prompt was used for each question:

```
You are a financial analyst. Extract data from the provided text
and tables to generate a step-by-step mathematical reasoning pro-
gram and the final numerical answer. Be precise and do not hallu-
cinate figures.
Context:
{pretext}
{table}
{posttest}
Question:
{question}
Calculation and Answer:
```

3.2.3 Code generation tasks

Software engineering (SE) is the field of development, implementation, and maintenance of software systems. It is one of the areas that benefits from the LLM revolution (Ma et al., 2024). LLMs are significantly changing the field of SE. LLM's have shown potential is handling complex tasks, can leave a remarkable effect on the SE practices and tools (Hou et al., 2024).

SWE-bench is a benchmark that contains GitHub issues from famous repositories that report bugs or request new features. The pull requests that actually merged to the source code resolving these issues are also added to the dataset as the solutions of the issues(Jimenez et al., 2024). Figure 4 shows the procedure of collecting useful issues and pull requests. Repositories may be noisy because of poor maintenance, documentation or ad-hoc fixes. So, they conducted the following process to extract high quality repositories.

Figure 4 Process of extracting repositories from GitHub (Jimenez et al., 2024, p. 2)

The content of each issue was used in the following prompt to pass it to LLM:

```
You are an expert software engineer. Analyze the repository to
locate the bug and provide the complete, corrected code for the
affected files. Ensure the solution is functional and passes all
tests.
```

```
Repo:
{repo}
Issue:
{issue}
Solution:
```

3.3 Deploying LLMs in the local environment

3.3.1 Llama.cpp and Ollama

Large language models were originally designed to run on cloud-based services with powerful computational infrastructure. However, due to growing concerns about data privacy, operational costs and carbon emissions have increased interest in deploying LLMs locally on institutional or personal hardware. Local deployment allows organizations to maintain control over sensitive data, avoid recurring cloud API fees and operate without internet connectivity. Two key technologies have made local LLM deployment practical: **llama.cpp** as the core inference engine and Ollama as the user-friendly deployment platform built on top of it. Understanding these frameworks is important for measuring and comparing the energy consumption of different LLM configurations in real-world settings.

llama.cpp is an open-source inference engine designed to run large language models efficiently on local hardware. The main goal of llama.cpp is to enable LLM inference with minimal setup and state-of-the-art performance on a wide range of hardware configurations (*LLaMa.CPP*, 2025). The framework was initially developed to support Meta's LLaMA models but has since expanded to support many model architectures. LLaMA introduced a collection of foundation language models ranging from 7B to 65B parameters, trained on trillions of tokens using publicly available datasets exclusively (Touvron et al., 2023). llama.cpp enables these models to run on consumer hardware through several technical optimizations. Ollama is an open-source platform that built on llama.cpp to make local LLM deployment accessible to non-expert users. Ollama is designed to simplify the deployment and management of large language models on local hardware, addressing key challenges developers and researchers face when using LLMs, such as cost, complexity and privacy.

3.3.2 Installation and setup

Ollama installation file is accessible from their website for Windows and for Linux operating systems the following commands need to be run:

```
curl -fsSL https://ollama.com/install.sh | sh
```

It will download and install Ollama on the machine. Ollama will serve as a running service on the OS and can be called using command line or from the official Python library called Ollama.

Researchers used the official Python library to pull, run and manage the following models available on the Ollama website. In Table 2 the list of the models with their features can be found. This table contains model names, their number of parameters, Size of the models, total number of downloaded and the capability of thinking of the model. At this research, the most popular and downloaded open-source models based on the number of downloads on Ollama website were collected.

Table 2 List of models used in this research

Model Name	Parameters	Size (Approx)	Downloads (Ollama Est.)	Thinking Available?
DeepSeek-R1:14B	14B	9.0 GB	82M+	Yes
DeepSeek-R1:7B	7B	4.7 GB	82M+	Yes
DeepSeek-R1:1.5B	1.5B	1.1 GB	82M+	Yes
Gemma 4:e4B	4B (Eff.)	3.2 GB	~5M (New)	Yes
Gemma 4:e2B	2B (Eff.)	1.8 GB	~5M (New)	Yes
Gemma 3:12B	12B	8.2 GB	35M+	No
Gemma 3:4B	4B	2.8 GB	35M+	No
Gemma 3:1B	1B	800 MB	35M+	No
CodeGemma:7B	7B	5.0 GB	12M+	No
CodeGemma:2B	2B	1.5 GB	12M+	No
Llama 3.2:3B	3B	2.0 GB	65M+	No
Llama 3.2:1B	1B	1.3 GB	65M+	No
Mistral:7B	7B	4.1 GB	28M+	No
DeepSeek-R1:70B	70B	42 GB	82M+	Yes
Llama 3.3:70B	70B	40 GB	113M+	No
DeepSeek-R1:32B	32B	20 GB	82M+	Yes
Gemma 4:31B	31B	19 GB	~5M (New)	Yes
Gemma 3:27B	27B	16 GB	35M+	No

Thinking availability is related to chain of thought architecture which models use during inference. Chain of thoughts is a series of intermediate reasoning steps that significantly improves the ability of LLMs to perform complex reasoning(Wei et al., 2023). Thinking availability results in producing more output tokens but improves the ability to solve complex problems that need to be explained step by step.

3.3.3 Inference procedure

To increase the speed of recording energy consumption during inference of LLMs, a Python script has been written to select the following parameters as the input:

- Benchmark name and file path
- Questions of the benchmark
- Model name
- Keep alive option
- Thinking argument

Keep alive argument refers to the capability of Ollama platform which keeps the model alive in the VRAM instead of dropping it. This feature allows users to increase the speed of their interaction with the models. This capability is considered as a feature to measure energy consumption in 2 phases, setup and running.

The Python script returns the following items as the output and stores them in a logging file:

- Model information
- Thinking capability True/False
- Keep alive True/False
- Number of Input Tokens
- Number of output Tokens
- Energy consumed (KWh)

3.4 Measuring electricity consumption

To accurately measure the energy consumption of the inference of LLMs, there was a need to monitor and collect data from multiple sources of energy consumption including RAM, CPU, and GPU. Each manufacturer of the hardware typically has their own software or tool to monitor the energy consumption of their products. Intel RAPL is the package for measuring energy consumption for Intel CPUs and RAM. GPUs from NVIDIA uses NVIDIA management library (NVML) which is an API provided by NVIDIA to monitor and manage various states of NVIDIA GPU devices. In this research, CodeCarbon library was used to measure energy consumption. This library is open source and uses the above-mentioned tools to measure and monitor the utilization rate, power and energy consumption of CPU, GPU and RAM. It has a built-in function to measure CO2 emissions based on energy consumption. It measures the power and energy consumed during the inference and returns all the required measurements for this research.

3.5 Business cost analysis

3.5.1 Business costs from employee's approach

To help Decision makers integrate electricity costs of running Inference on LMM's into their operational expenditures (OpEx) a new estimation framework was developed. In this estimation framework, the following standard ranges for company size that was used are as follows (Honkanen, 2022). This categorization which can be seen on Table 3 was selected because it divides the companies into multiple sections based on their number of employees. It was used for estimation of the total time each company (based on their number of employees) spent on using large language models for their daily tasks.

Table 3 Company categories by the number of their employees

Company Size	Number of Employees
Micro	1 to 10
Small	11 to 50
Medium	51 to 200
Large	201 to 1000
Enterprise	1000+

According to (Alexander Bick et al., 2025), among people who reported using generative AI during last month at least one time, 31.9% said they are using it 60+ minutes per day (Usual users), 47% reported usage between 15 – 59 minutes per day (Regular users) and 21% reported under 14 minutes per day (Rare Users). Figure 5 shows the summary of the survey conducted by Alexander Bick et al. concluding the percentage of employees who use AI for daily business tasks.

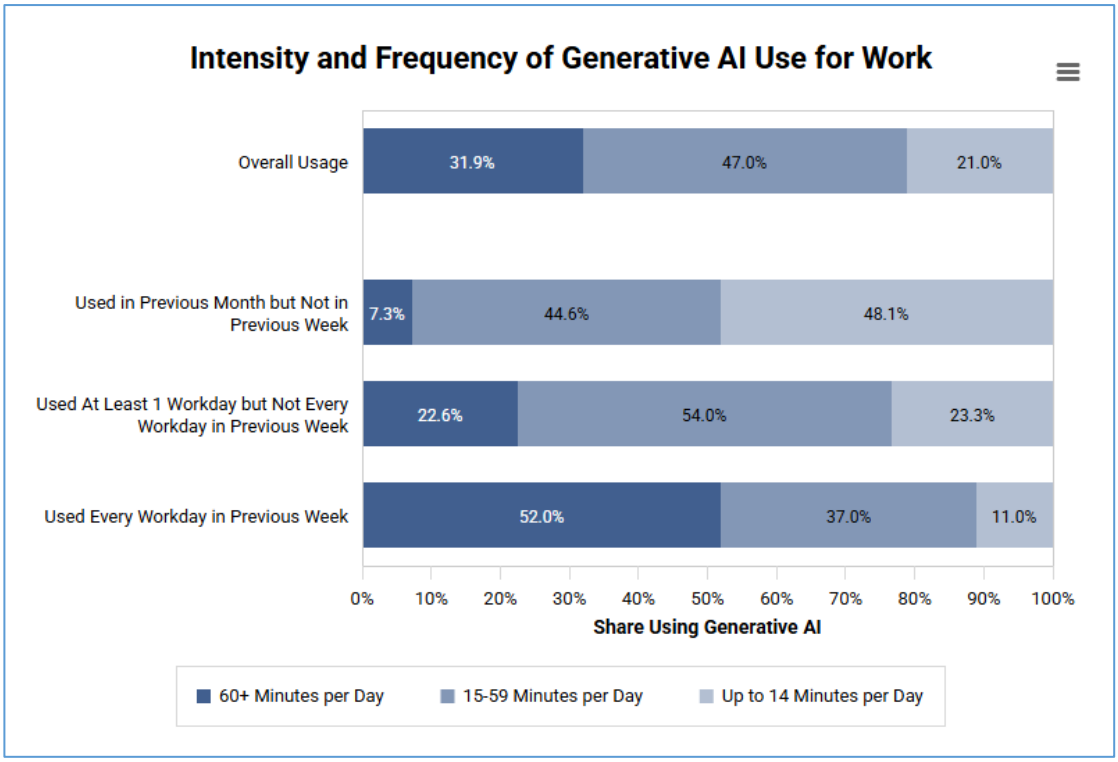

Figure 5 Usage of generative AI during workday (Alexander Bick et al., 2025)

These rates were used for estimating the total hour of LLM usage in different sizes of companies using the following formula:

$$LLM\ usage = \sum rates \times minutes \times n \quad (1)$$

Where n represents the number of employees the company has. Replacing the rates and number of minutes (approximately) gives the value which can be used to estimate the total hours each company uses generative AI for their tasks per day. For usual users, the total hours per day was more than 60, this number was estimated as 65 minutes per day. For regular and rare users, the average of their upper and lower limits was selected (37.5 and 7.5 minutes per day).

On the other hand, this research classifies business tasks are three segments: 1. Summarization, 2. Coding, 3. Numerical Analysis task. So, the calculations to estimate the total energy bill of the company were done by using the average of these 3 benchmarks.

According to section 3.4 the output for the experiments returned number of output tokens and the duration of the process as well as the energy consumption. Two required KPI for estimating

energy bills were achieved by these measurements. The first one calculated by the following formula gives the Energy consumption per token (EPT):

$$EPT = \frac{E\ (Kwh)}{N_o} \tag{2}$$

Where E stands for Energy consumption and N_o represents the number of output tokens. The second KPI shows the number of tokens that can be generated by the model per second (TPS) and calculated using the following formula:

$$TPS = \frac{N_o}{D} \tag{3}$$

Where D represents the duration of the inference of the model (in seconds). Multiplying these two KPI by each gives us a unit of power of different models and it is calculated for different models answering different benchmarks. Using the following formula total energy consumption of a company using each model and each benchmark can be calculated:

$$E_{tot} = EPT * TPS * t_u \tag{4}$$

Where t_u is the time of each company uses AI in their daily tasks.

3.5.2 Comparison costs per 1M tokens

Current industrial standards for pricing online LLM APIs are by giving the cost per 1 million tokens. Usually input tokens are 3 to 5 times cheaper than output tokens. To make a reliable comparison, the costs per 1 million tokens were obtained from the following formula:

$$Cost\ per\ 1M\ tokens = \frac{E}{N_O + N_I} \times 1{,}000{,}000 \tag{5}$$

Where E represents the average energy consumed per each prompt from the benchmarks and $N_O + N_I$ is the number of Input plus Output tokens. This value was calculated for thinking capability being On or Off separately.

The price for the Online LLM APIs was collected from a benchmark website called helicone.ai. Figure 6 shows a screenshot from this website.

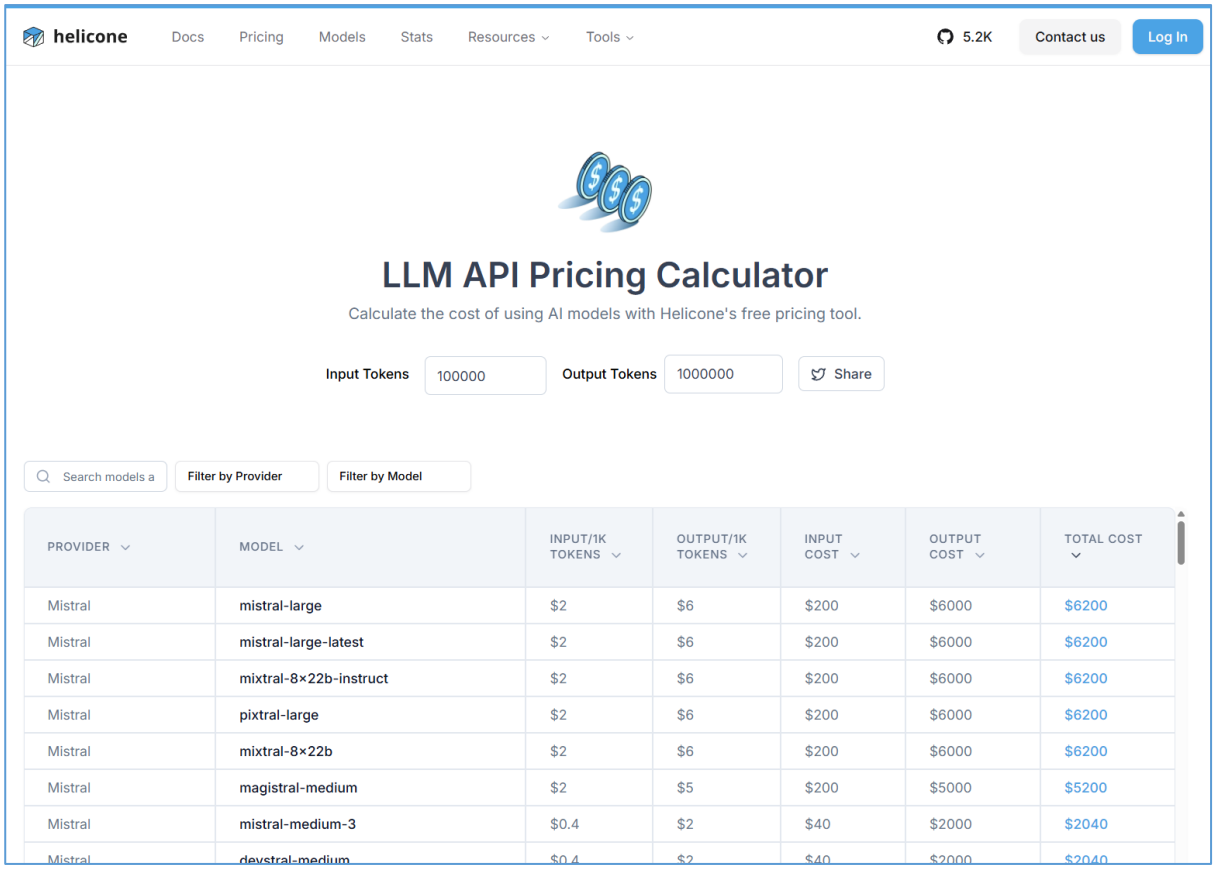

Figure 6 Helicone website for collecting Online LLM API pricing

It shows all the online services divided by the name of their providers, model name and their prices. Most common models and their providers are OpenAI, Google, Anthropic and Mistral. Their products are usually closed source, and the number of their parameters are not visible to usual users. But services like OpenRouter.ai allows using open-source models like Deepseek-r1 as well. Therefore, the price of open-source models which were used in these experiments was collected from the openrouter provider on helicone website.

Another issue to consider was the side costs of running LLMs which were not calculated in this research's measurement but need to be considered since online APIs will consider it in their pricing. According to API unit economic models by FutureSearch (Dan Schwarz et al., 2025), commercial AI providers typically works with gross margin between 50% and 75%. Therefore, half to three quarters of commercial AI pricing relates to profit. This conclusion can be seen on Figure 7.

Figure 7 OpenAI API Profit (Dan Schwarz et al., 2025)

ICONIQ's January 2026 State of AI report finds inference costs of an AI product by roughly account for 25% of the total costs (ICONIQPartners, 2026). So, the rest of 75 % of costs come from all other costs including Talent costs, infrastructure and cloud, data storage and processing, model training and compliance. So, by multiplying the local LLM inference cost value by 4 we can get a rough approximation of the total costs for running the models locally. As can be seen on Figure 8, 77% of AI product costs will be used for other additional costs of the product.

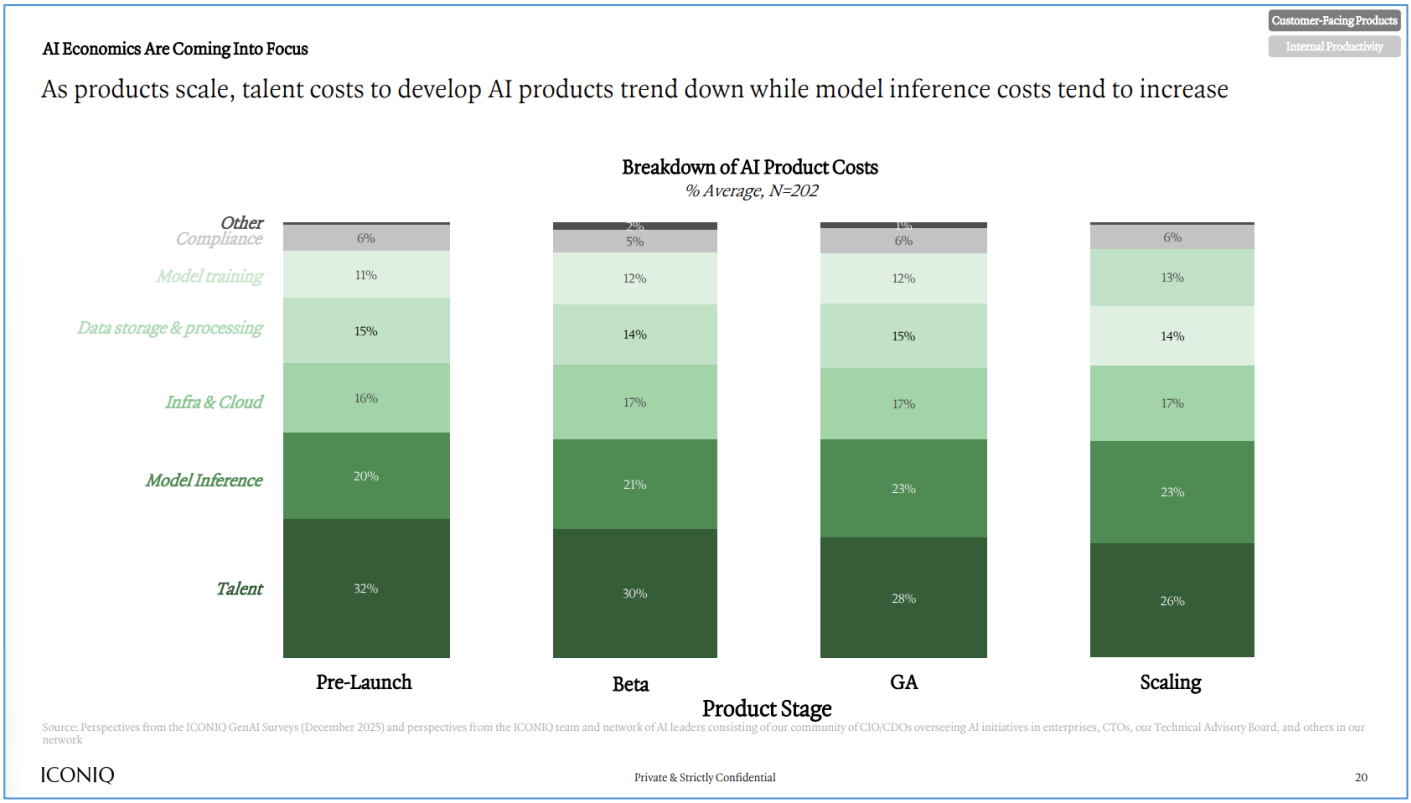

Figure 8 AI Product costs (ICONIQ, 2025)

Therefore, to have a reasonable value to compare locally deployed LLM costs with Online LLM APIs, the energy costs of locally deployed LLM were multiplied by 4 and compared to Online LLM API prices.

$$C_{tot}^{L} = 4C_I \tag{6}$$

Where C_{tot}^{L} is local LLM costs, and C_I is the cost of inference.

3.6 Summary

In this chapter, the methodology of this research and the procedures through which experiments were conducted were explained. The infrastructure and benchmarks utilized were detailed in Sections 1 and 2. The instructions for deploying Large Language Models (LLMs) on local hardware were outlined in Section 3, and the specific energy measurement tools employed were identified in Section 4. Furthermore, the methodology for business cost analysis was established, and the processes by which results were applied to real-world business cases are explained in detail.

4 Results

4.1 Visualizations and comparisons

This section includes visualizations from the results obtained from the experiments. It is divided into 4 sections including the direct results of the experiment which contains the chart for number of output tokens per model and benchmark and average energy per token per model and benchmark. A comparison of previous research is included to verify the results obtained. In the last two sections, the result for two main methodologies is presented. The first one calculating business costs from employees' approach and the second is the Cost per 1M comparison.

4.1.1 Experiment Results

There were multiple KPIs to consider and monitor. First are the questions and benchmarks. There were 10 questions per benchmark which were chosen randomly but they remained the same during the inferences with different models. For Heavy models (Deepseek-r1:70b, Llama3.3:70b, DeepSeek-R1:32B, Gemma 4:31B, Gemma 3:27B) to increase the speed of data collection the first 5 questions of the selected question were used.

Questions will break down to tokens by the model and because of the different architectures of training of different models, the input tokens will be different for each model (Lotz et al., 2025, p. 2).

In Figure 9 the average number of input tokens grouped by each model is shown. As can be seen, the model Gemma3:27b and Gemma4:31b tokenizes the SWE-bench prompts to higher number of tokens than other models and benchmarks while Codegemma:2b breaks it down to half of others.

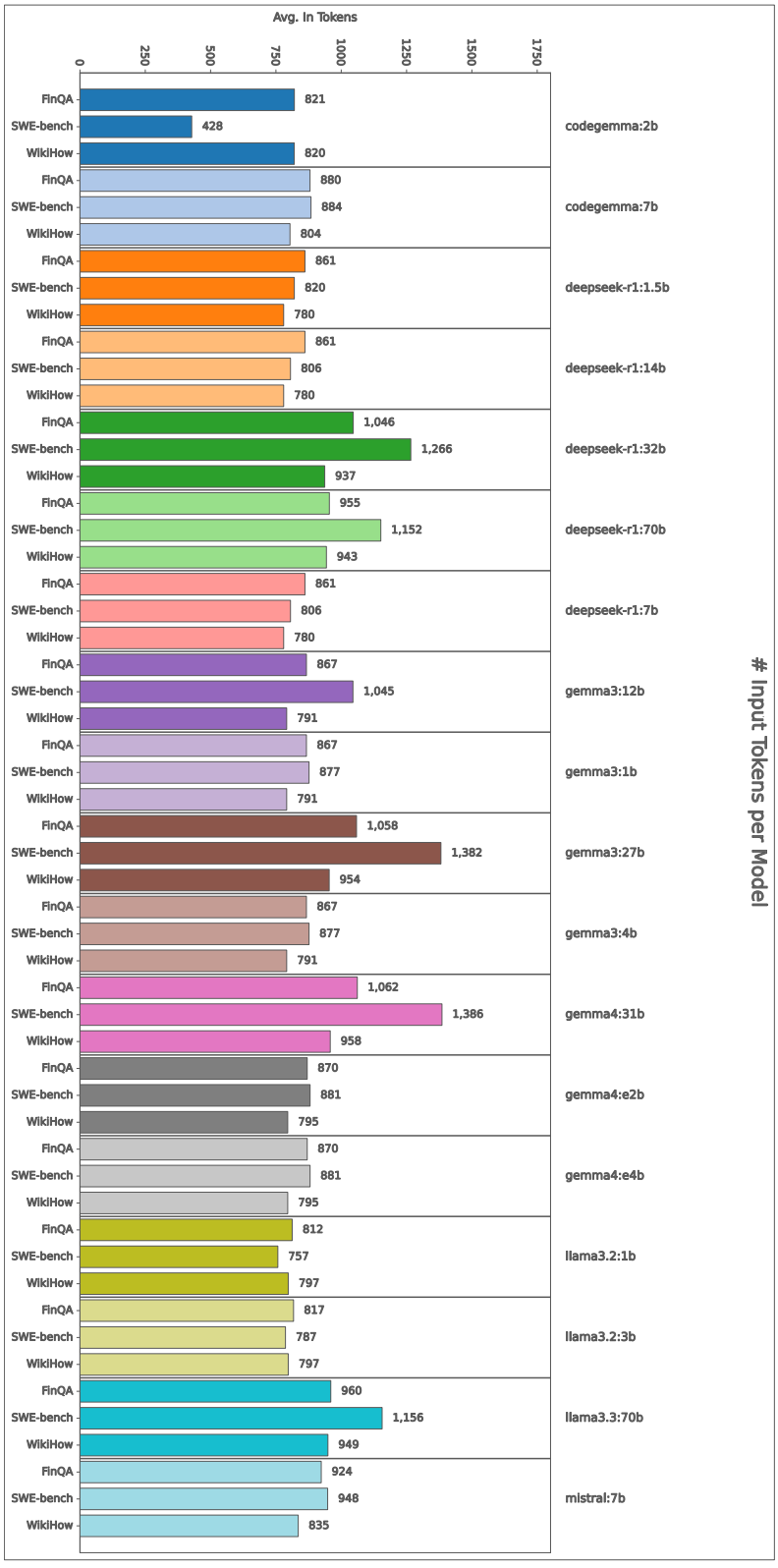

Figure 9 Number of input tokens per model

Number of input tokens per benchmark are roughly the same magnitude as shown in Figure 10. By average FinQA had 873.25, SWE-bench had 870.52, and WikiHow had 806.54 tokens per prompt for their inputs.

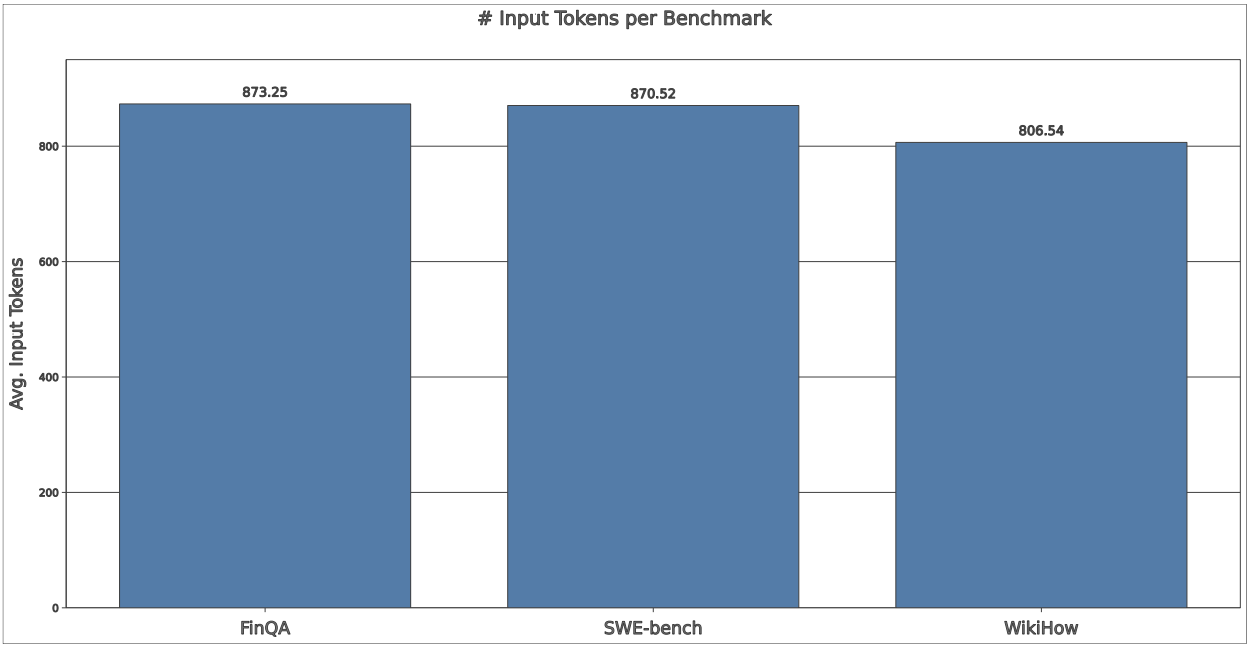

Figure 10 Average number of input token per benchmark

Figure 11 showing that answering FinQA benchmark questions generates more Tokens compared to WikiHow is the lowest one when thinking capability is On (considering that WikiHow benchmark was used to generate summarizations based on the context of the input). An important finding is comparing FinQA output tokens while thinking is On and Off. With thinking being On it generates tokens more than other benchmarks but while it is Off, it ranks the lowest output token generator. Answering FinQA benchmark questions generates 492 tokens by average when thinking is Off and while it is On, it generates 2642 tokens. For SWE-bench answering with thinking On generated 2113 by average, and 734 when thinking was Off. WikiHow benchmark generated 497 and 1187 while thinking was Off and On respectively.

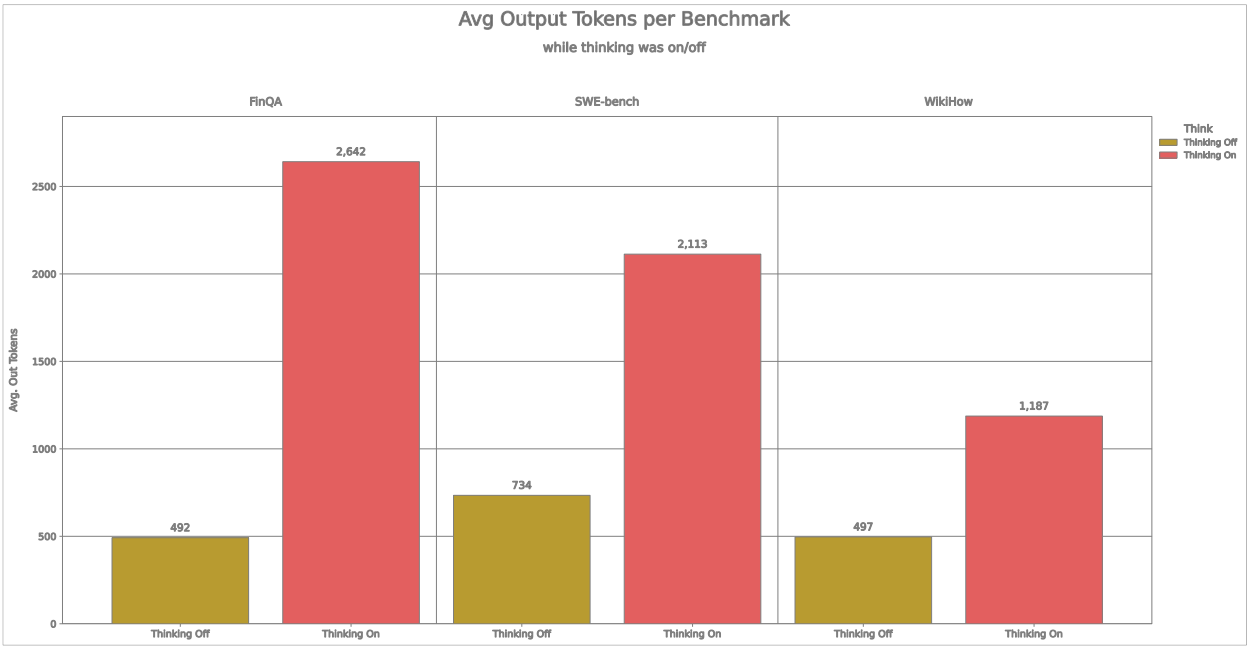

Figure 11 Average Number of output tokens per benchmark with Thinking ON/OFF

Number of output tokens per model follows the same pattern. It varies for Thinking capability being On or Off. Figure 11 shows the average number of output tokens for each model while thinking was Off. There is a significant variation between the number of output tokens per model. While Codegemma:2b generated 2009 tokens per question for the WikiHow benchmark and 1118 and 1041 tokens per question for the SWE-bench and FinQA respectively, Codegemma:7b generated 242, 330, and 270 for WikiHow, SWE-bench, and FinQA respectively. Gemma4:31b generated 1302 tokens per question for SWE-benchmark while its output for FinQA and WikiHow is 550 and 507 respectively.

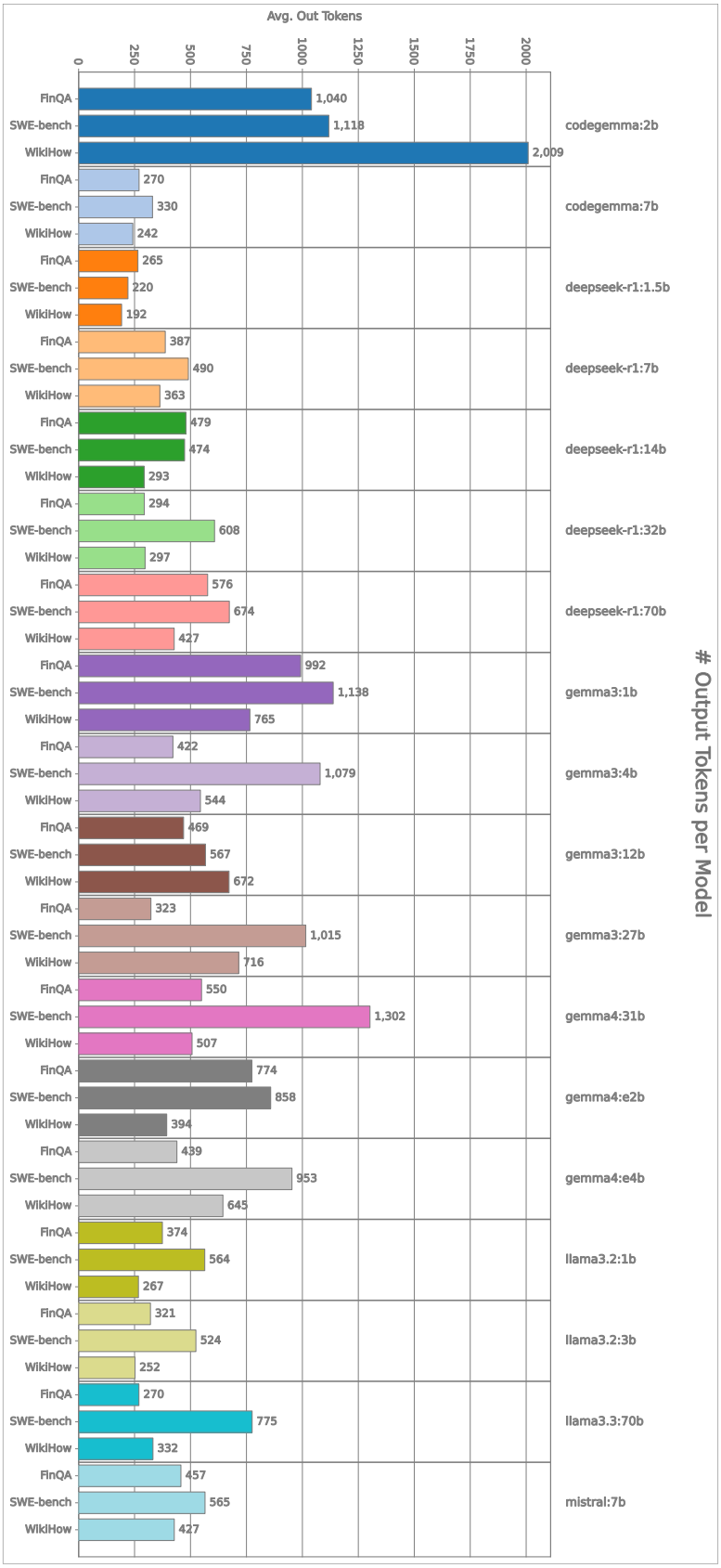

Figure 12 Number of output tokens per model with Thinking Parameter set to OFF

Thinking Capability is not available for all models, therefore the chart for the number of output tokens per model while thinking was on is different than Figure 12. Gemma4:31b shows a significant out of range number of tokens to answering the FinQA and SWE benchmarks, which are the analytics tasks, while answering the WikiHow benchmark is in the order of other models. Deepseek-r1:1.5b takes generates 3 times more tokens for answering FinQA than SWE. Figure 13 shows the difference between number of output tokens per model while thinking was On.

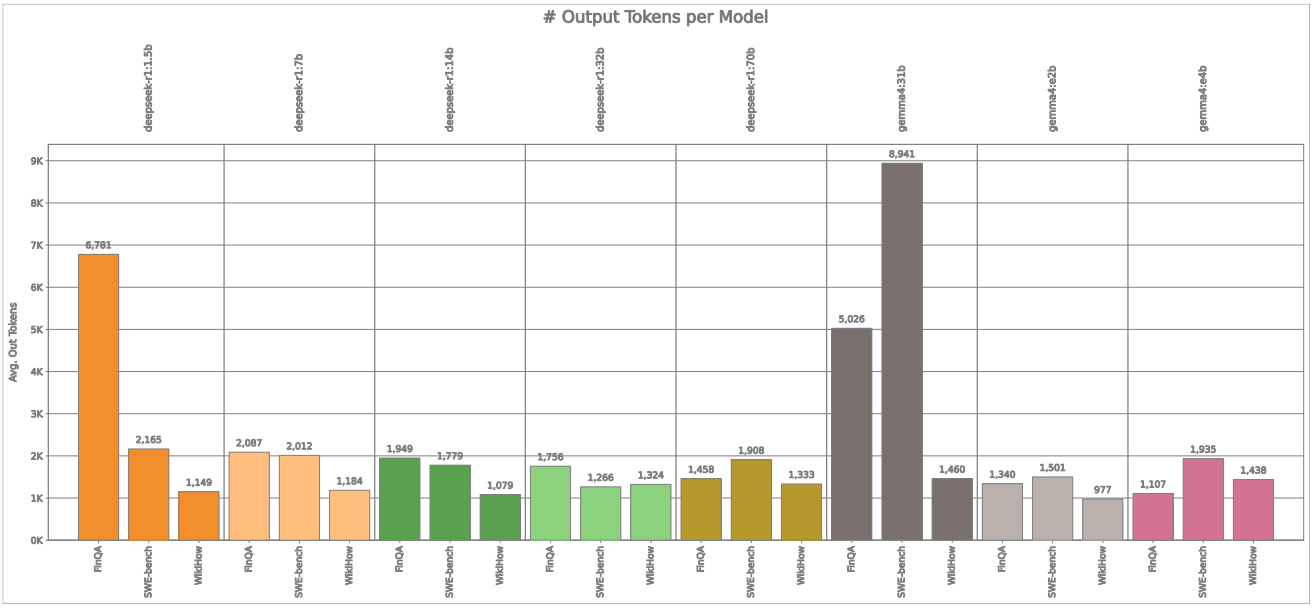

Figure 13 Number of output tokens per model with Thinking parameter set to ON

Figure 14 shows a better view of the average number of outputs being generated by models by showing the thinking capability being On or Off. Deepseek-r1 family models show a similar behavior. While thinking was Off, they generate almost half tokens than other models but with thinking being ON they generate 3 times more tokens. Gemma4:31b, which is a new and heavy model by Google, ranked as the first model in generating tokens while thinking is On by 5K tokens.

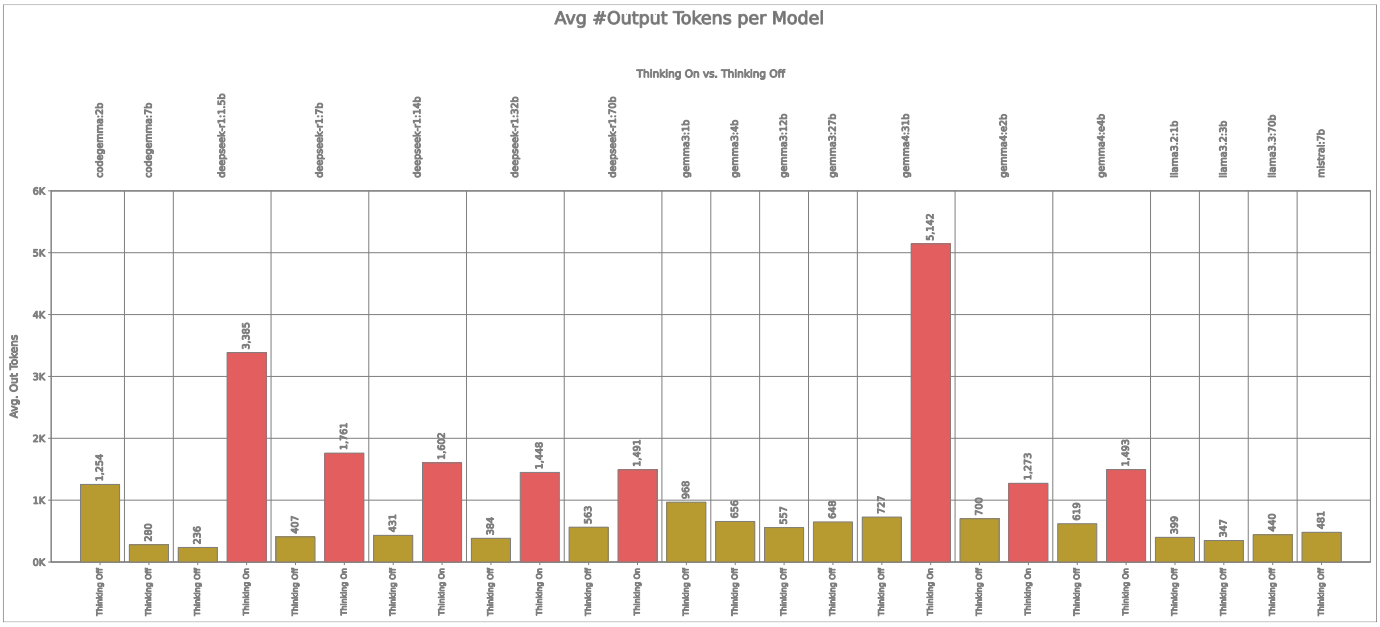

Figure 14 Number of output tokens by models

As the token per second (TPS) metric will be used in the estimation of business costs from employee's perspective, the speeds of the models will be evaluated in the Figure 15. Model sizes show a major correlation with speed of inference of models. Gemma3:1b with 170.1 token per second ranked as the fastest model while with a slight difference, Codegemma:2b with 167.8 tokens per second ranked the second. Deepseek-r1:1.5b showed half slower than other models with 86.6 tokens per second while llama3.2:1b with 117.5 tokens per second was slower than its peer models. Middle size models like deepseek-r1:7b, Gemma3:4b, Gemma4:e4b and mistral:7b was in the same range with 72.7, 88.3, 77.1, and 83.3 tokens per second respectively. Larger models like deepseek-r1:32b, Gemma3:27b, Gemma4:31b were slower than middle sized models with 11.5, 23.8, and 23.2 token per seconds respectively. Heavy models like deepseek-r1:70b and llama3.3:70b with 3.9 and 3.8 token per seconds were the slowest models of this experiment.

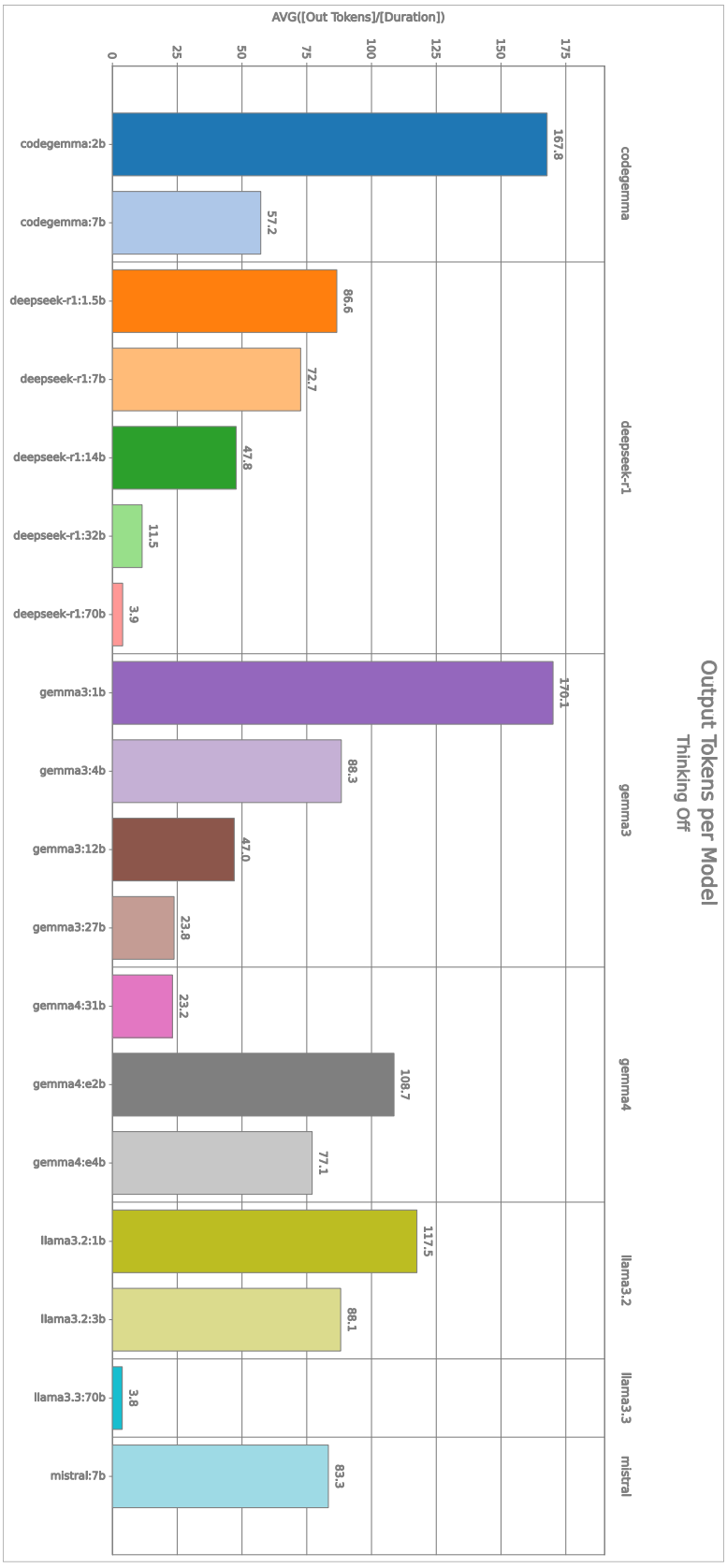

Figure 15 Average Token per seconds per model, thinking Off

For models with thinking availability being On, the same conclusion is valid for DeepSeek-r1 family. Deepseek-r1:1.5b generated 191 tokens per second and for the 7b, 14b, 32b, and 70b models it shows a decreasing trend with 98.9, 58.6, 12.9, and 4.1 token per seconds respectively. Compared to Gemma4 family, Gemma4:32b was faster than its deepseek-r1 peer and generated 27.8 tokens per second, while Gemma4:e2b and Gemma4:e4b was 126.4 and 96.4 tokens per second respectively. Figure 16 shows a comparison of average token per second for models with thinking capability being On.

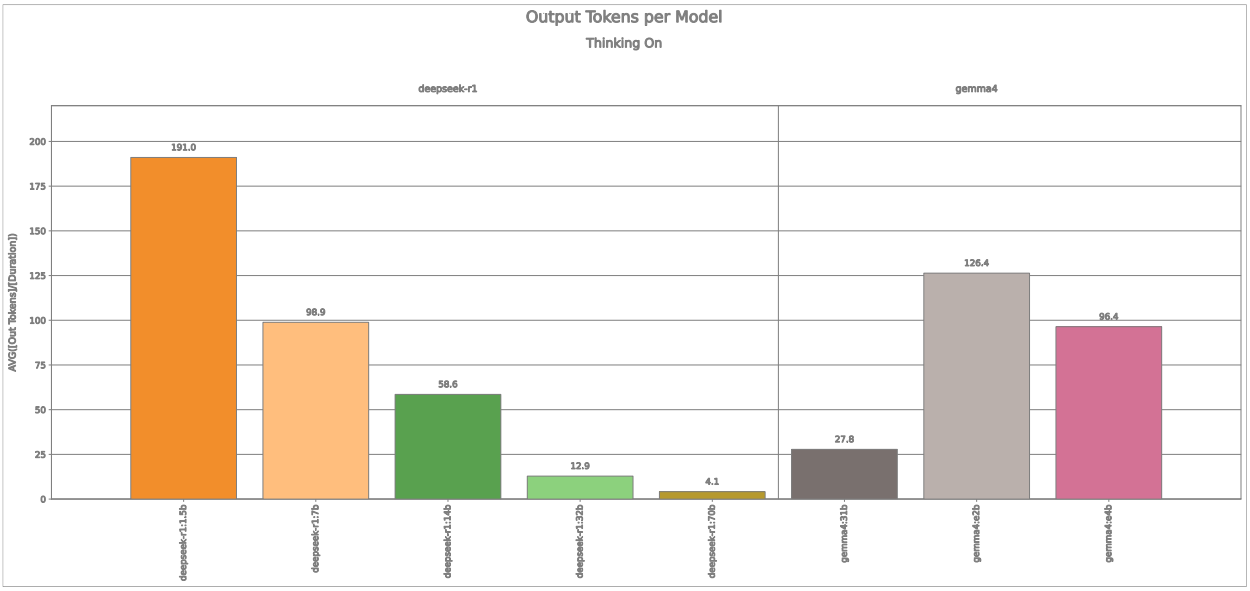

Figure 16 avg token per second for models, thinking On

In terms of energy consumption, the metric of Energy per token was measured and the average of these metrics is reported below. `Thinking capability` and `Keep-alive` option affected the Energy per token significantly, so for each combination the chart will be different. Figure 17, shows the Average energy per token for each benchmark in 4 different scenarios, considering thinking capability On or Off and keeping models in the memory for next inference or dropping them. Answering FinQA questions with thinking On and not keeping the models in the memory, used 2.5e-6 KWh energy and ranked as the most energy consuming scenario, while in the same situation answering SWE-bench with 1.5e-6 ranked as the least energy consuming scenario.

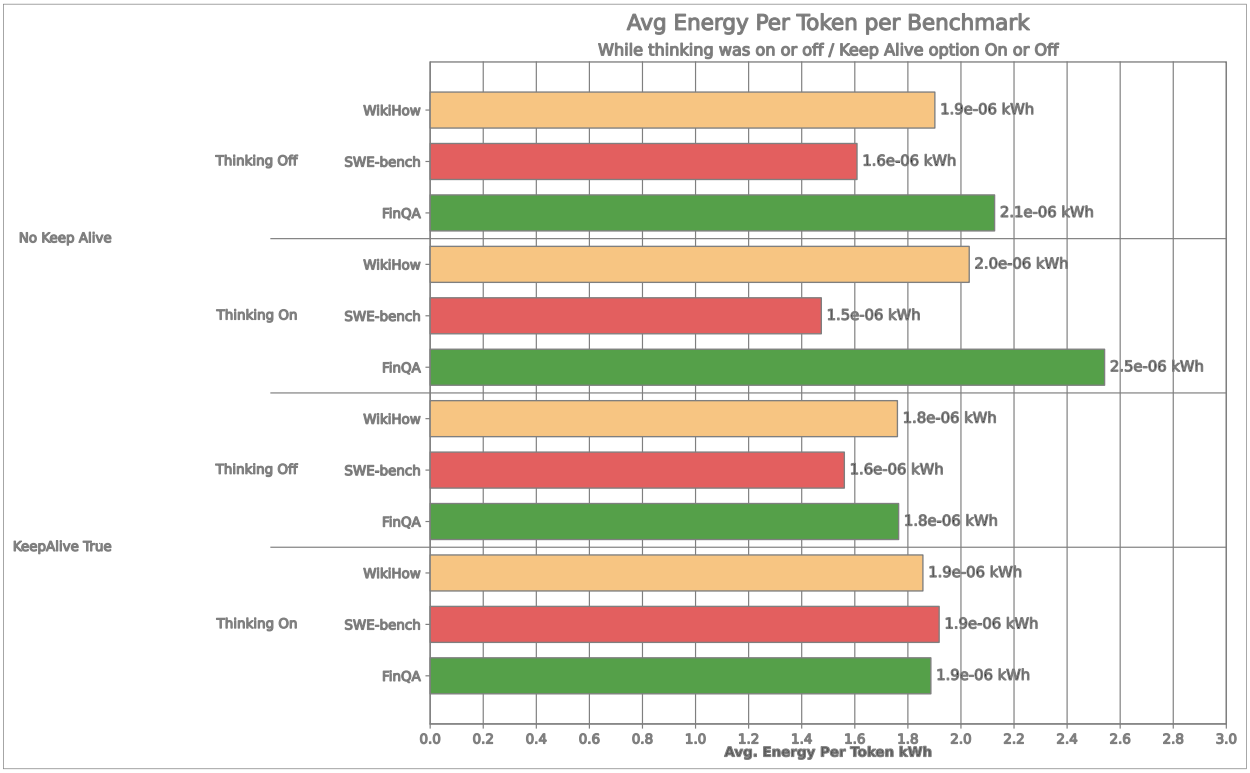

Figure 17 Energy per token per benchmark, considering thinking and keeping alive option

Average energy per token per model can be seen in Figure 18. For thinking capability being Off and keeping models in memory, Deepseek-r1:70b and llama3:70b model stands as the most energy consuming models with 1.7e-5 kWh per token. Afterward middle range models like deepseek-r1:32b, Gemma3:27b, and Gemma4:31b are ranked as the second most energy consuming models with 6.6e-6 and 4.6e-6 kWh respectively. Other smaller models (e.g. deepseek-r1:7b, Gemma3:4b, llama3.2:3b, etc.) consumed smaller amount of energy per token than others. Not keeping models in memory adds a slight amount of energy to the mentioned values. Llama3:70b and deepseek-r1:70b consumes 0.2e-5 and 0.1e-5 kWh more energy respectively.

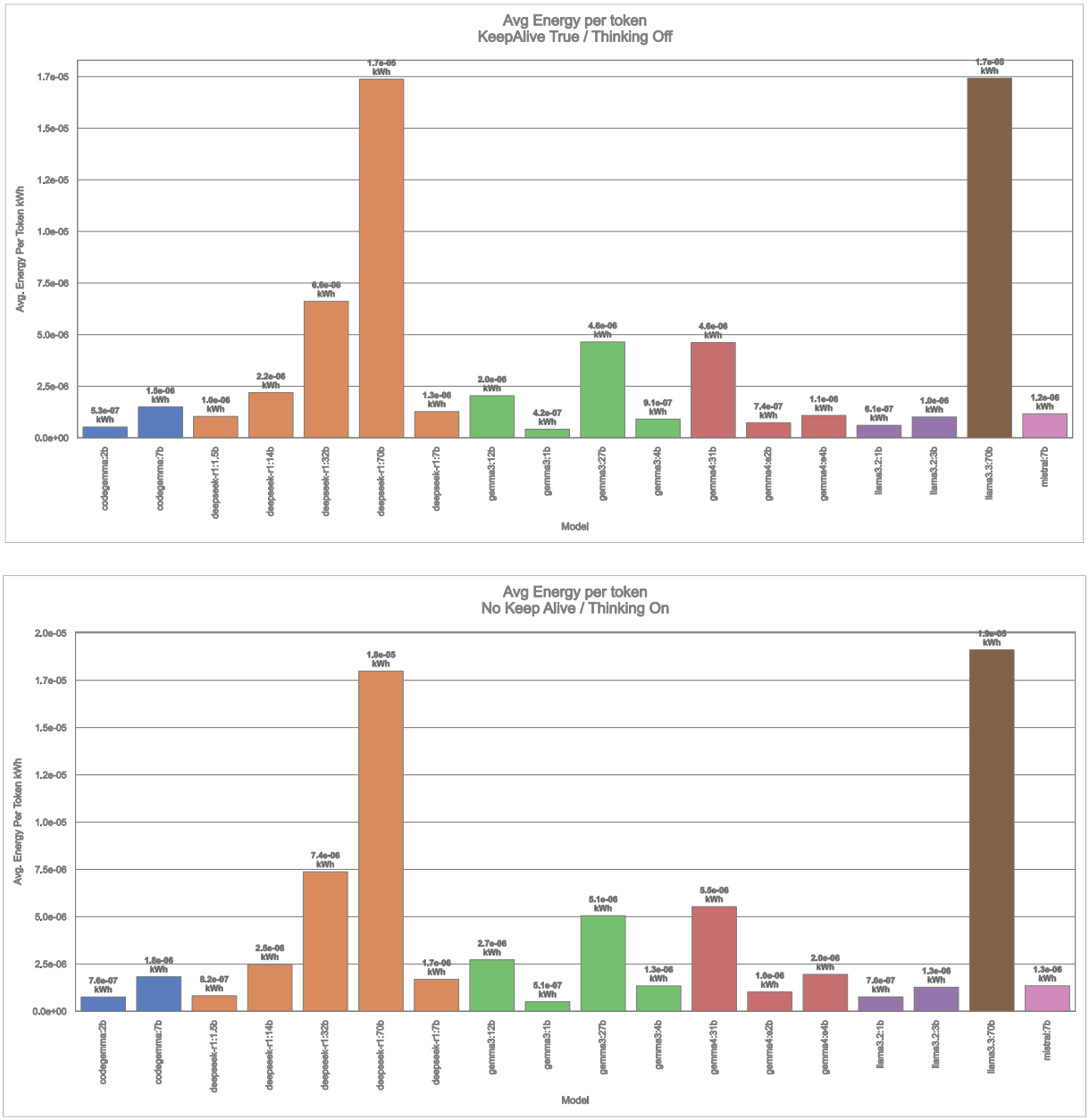

Figure 18 Average Energy per token per model, Thinking Off

Among models that could turn Thinking capability On and Off, Deepseek-r1:70b with 1.7e-5 KWh per token, ranked the first place of energy consumption. Deepseek-r1:32b and Gemma4:31b consumed 6.1e-6 and 4.5e-6 respectively and spot ranks 2 and 3 while thinking was On and not keeping models in the memory. Figure 19 shows the comparison of energy consumption of models per token while Thinking was On.

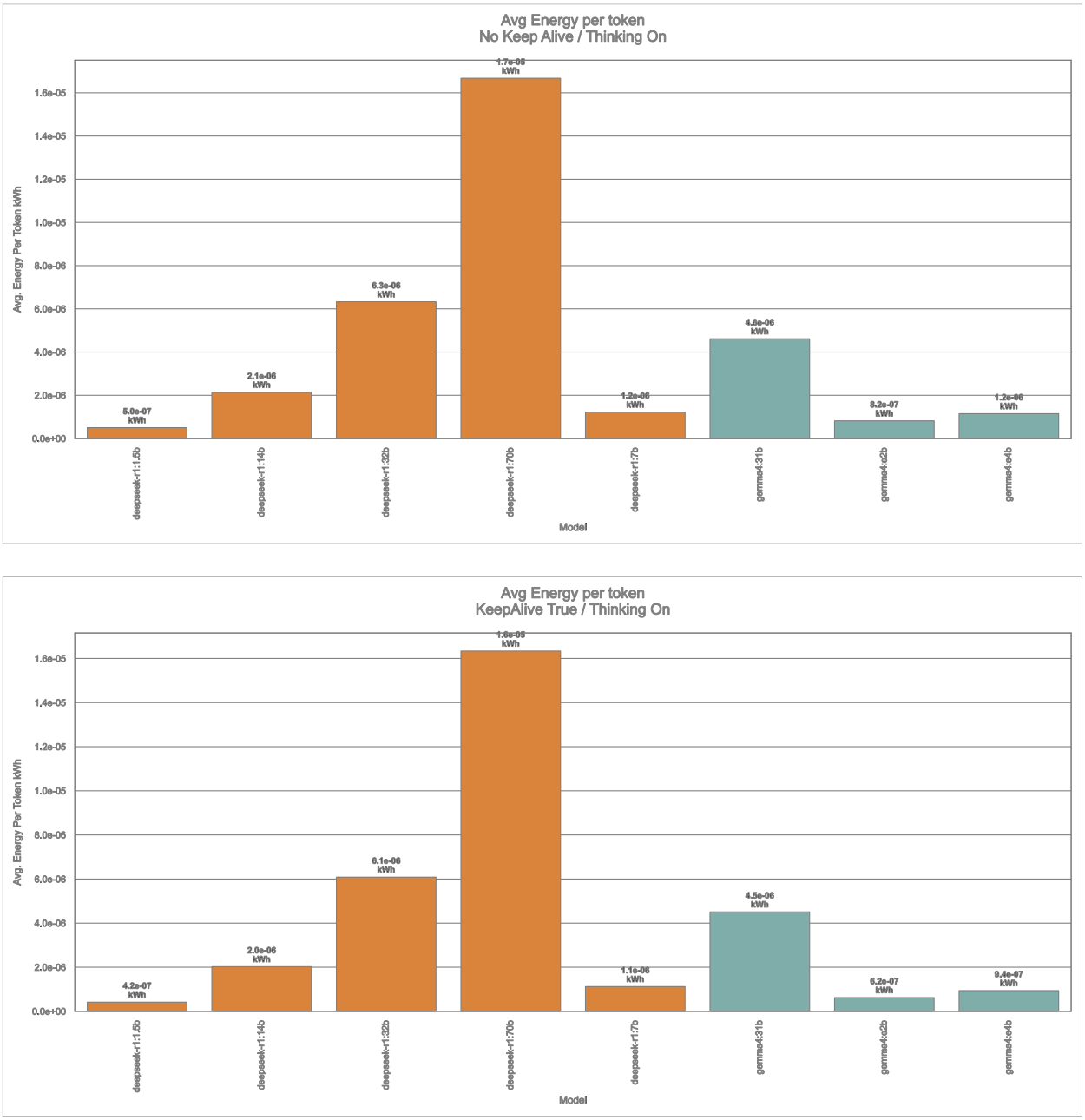

Figure 19 Average energy per token per model, Thinking on

4.1.2 Comparison to Previous research

Regarding comparing the achieved results with similar papers, the results from the research by Husom et al. were used (Husom et al., 2026, p. 5). Evaluated models divided into 3 groups based on the size of their parameters. They run the models locally on a server using NVIDIA ETX A5000 GPU with 24 GB and 2 laptops. One, with an NVIDIA Quadro RTX 4000 GPU with 8 GB and the other one using an Intel i5 11th Gen with 12 cores of CPU with 16GB of memory. In the Figure 20 the average of the energy consumed per token per model was used. As can be seen, the results

are in the range of their paper, and this matter can be used as an evaluation of this study results. The difference in energy consumption can be explained by the differences in the benchmarks used, as each benchmark applies to a distinct set of prompts and tasks that influence the overall energy consumption of the model but the patterns of larger models with higher parameters having higher energy consumption hold true.

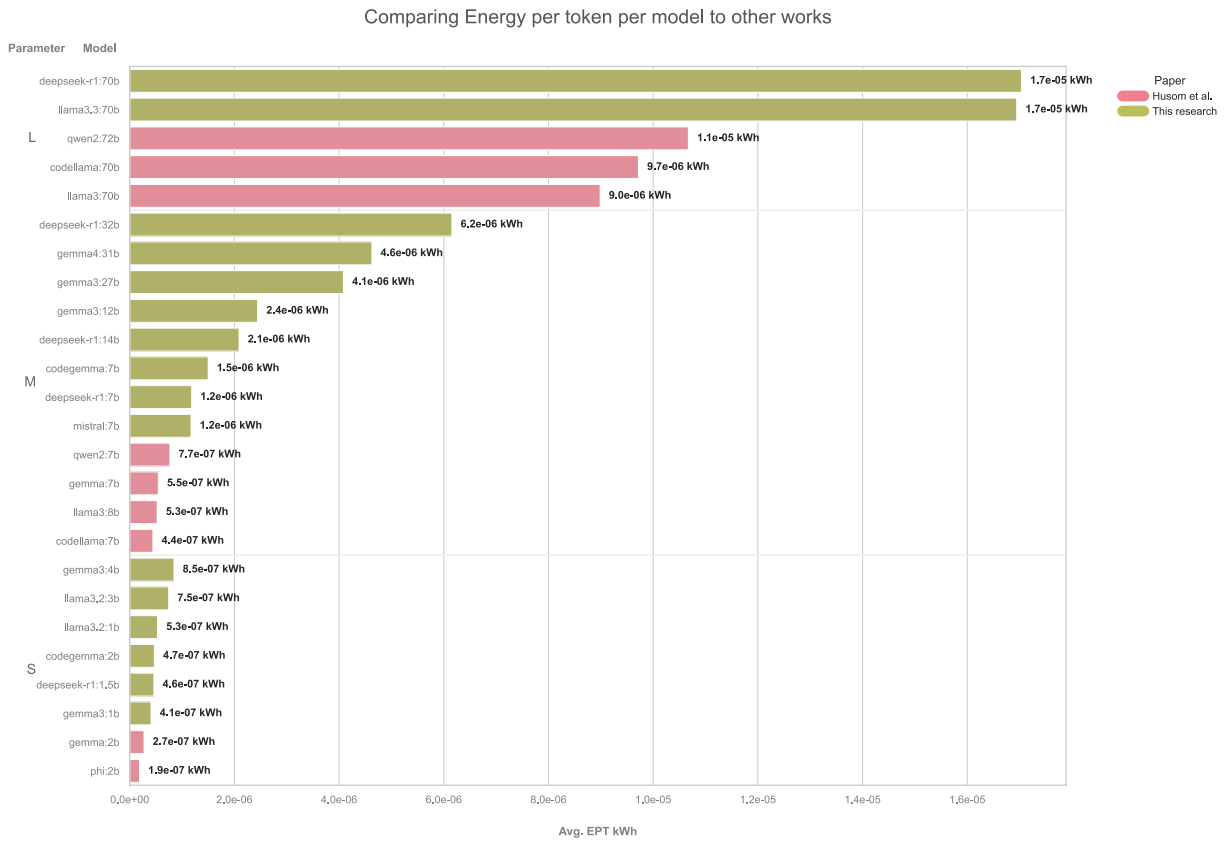

Figure 20 Comparing average energy per token with Husom et al. paper

4.1.3 Business calculation results from employees' approach

Table 4 shows the final calculation for estimating the total hours employees of a company use LLM to do their tasks. For usual users, who are the 31.9% of the employees an estimation of 65 minutes per day were used for calculation. Regular users, with 47% of the employees, took 37.5 minutes per day, which is an average of 15 and 59 minutes range mentioned in the paper. Rare users, with 21% of the employees, took 7.5 minutes per day as an average of up to 14 minutes per day for usage of AI to assist with their daily business tasks. The results of the calculations show that micro businesses with 1 to 10 employees uses AI for a total of 3.33 hours per day, while small businesses with 11 to 50 employees use 19.97 hours per day. For medium, large and enterprise businesses this number is 83.20, 399.35, and 998.38 hours per day in total.

Table 4 Estimation of total minutes of usage of AI in different companies

		LLM usage (min)	65	37.5	7.5	
		rate	0.319	0.47	0.21	
company size	number of employees	Avg number of em-ployees	Usual users	Regular users	Rare users	Total (Hours)
Micro	1 to 10	5	103.7	88.1	7.9	3.33
small	11 to 50	30	622.1	528.7	47.3	19.97
medium	51 to 200	125	2591.9	2203.1	196.9	83.20
large	201 to 1000	600	12441.0	10575.0	945.0	399.35
Enterprise	1000+	1500	31102.5	26437.5	2362.5	998.38

To have an estimate of the total energy consumption per token (EPT) and total number of tokens generated per second (TPS) the average for all three benchmarks were used as the final measurements. Table 5 shows the average values for EPT and TPS.

Table 5 TPS and EPT for all models and benchmarks

	FinQA		SWE-bench		WikiHow		Average	
Models	EPT (KWh)	TPS	EPT (KWh)	TPS	EPT (KWh)	TPS	EPT (KWh)	TPS
codegemma:2b	4.9E-07	188	4.7E-07	199	4.5E-07	218	4.7E-07	202
codegemma:7b	1.6E-06	55	1.5E-06	63	1.6E-06	54	1.6E-06	57
deepseek-r1:1.5b	4.9E-07	211	4.6E-07	199	4.6E-07	179	4.7E-07	196
deepseek-r1:14b	2.1E-06	58	2.1E-06	59	2.1E-06	56	2.1E-06	57
deepseek-r1:32b	6.3E-06	12	6.2E-06	13	6.1E-06	13	6.2E-06	13
deepseek-r1:70b	1.7E-05	4	1.7E-05	4	1.7E-05	4	1.7E-05	4
deepseek-r1:7b	1.2E-06	99	1.2E-06	101	1.2E-06	95	1.2E-06	98
gemma3:12b	2.3E-06	45	2.4E-06	40	2.1E-06	52	2.3E-06	46
gemma3:1b	4.3E-07	177	4.1E-07	191	4.6E-07	160	4.3E-07	176
gemma3:27b	5.1E-06	20	4.1E-06	30	4.2E-06	27	4.4E-06	26
gemma3:4b	1.1E-06	73	8.5E-07	112	1.0E-06	86	1.0E-06	90
gemma4:31b	4.6E-06	28	4.6E-06	28	4.5E-06	26	4.6E-06	27
gemma4:e2b	7.4E-07	116	6.9E-07	129	7.7E-07	108	7.3E-07	118
gemma4:e4b	1.2E-06	77	9.8E-07	102	1.0E-06	93	1.1E-06	91
llama3.2:1b	6.3E-07	112	5.3E-07	147	7.5E-07	93	6.4E-07	117
llama3.2:3b	9.4E-07	84	7.5E-07	120	1.0E-06	76	9.0E-07	93
llama3.3:70b	1.9E-05	4	1.7E-05	4	1.8E-05	4	1.8E-05	4
mistral:7b	1.2E-06	82	1.2E-06	91	1.2E-06	83	1.2E-06	85

Applying these values to the company sizes by the number of employees gives us the following table which shows the daily electricity cost estimates for working with LLMs deployed locally. The average electricity price was obtained from the Spot Price of energy from Fingrid website which is **0.04 €/kWh**. Applying the price and combining the company sizes table gives us the Table 6 which is monthly electricity bill of using locally deployed LLM by company employees divided by different models.

Table 6 Monthly Electricity cost estimates for using LLMs for different company sizes

			Energy cost by Company size [€/month]				
Models	Energy per hour (KWh)	Energy per hour (Euro)	Micro	Small	Medium	Large	Enter-prise
codegemma:2b	0.34	0.04	1.10	5.94	25.08	120.56	301.40
codegemma:7b	0.32	0.04	0.88	5.72	23.76	113.96	284.68
deepseek-r1:1.5b	0.33	0.04	0.88	5.94	24.42	116.82	291.94
deepseek-r1:14b	0.43	0.05	1.32	7.70	31.68	152.02	380.16
deepseek-r1:32b	0.29	0.03	0.88	5.06	20.90	100.54	251.46
deepseek-r1:70b	0.25	0.03	0.66	4.40	18.04	86.24	215.60
deepseek-r1:7b	0.42	0.05	1.32	7.26	30.58	146.52	366.30
gemma3:12b	0.37	0.04	1.10	6.60	27.28	131.12	327.58
gemma3:1b	0.27	0.03	0.88	4.84	20.02	96.14	240.02
gemma3:27b	0.41	0.05	1.10	7.26	30.14	144.32	361.02
gemma3:4b	0.32	0.04	0.88	5.72	23.76	113.74	284.46
gemma4:31b	0.45	0.05	1.32	7.92	32.78	157.74	394.24
gemma4:e2b	0.31	0.04	0.88	5.50	22.66	109.12	272.80
gemma4:e4b	0.35	0.04	1.10	6.16	25.52	122.98	307.34
llama3.2:1b	0.27	0.03	0.88	4.84	19.80	94.82	237.38
llama3.2:3b	0.30	0.04	0.88	5.28	22.22	106.26	265.76
llama3.3:70b	0.24	0.03	0.66	4.18	17.82	85.58	214.28
mistral:7b	0.37	0.04	1.10	6.38	27.06	129.80	324.28

From the table it is clear that electricity costs for running inference on a locally deployed model are almost negligible. So, the main cost consideration factor for local deployment of LLMS is not the electricity cost but costs associated with infrastructure and personnel that are needed to operate and maintain the locally deployed LLM system. To estimate the total cost of running the LLMs deployed locally and compare them with cloud computing we need to estimate costs in

terms of 1M tokens usage as the service providers often sell access to LLM services based on tokens instead of the Usage.

4.1.4 Costs per 1M token comparisons

Results for cost of 1 million tokens for each model are as follows. Figure 22 shows that Llama3.3:70b with 17.65 kWh energy per 1M token is the most energy consuming model followed by Deepseek-r1:70b which in both scenarios (thinking On or Off) consumes 16.84 and 16.64 kWh respectively. Among middle sized models, Deepseek-r1:32b consumed 6.69 kWh per 1M token while thinking was Off and 6.13 kWh while thinking was On. Other middle-sized models like Gemma4:31b and Gemma3:27b consumed almost the same amount of 4.60 and 4.31 kWh while thinking was Off. Generally, models consumed slightly less memory when their thinking was On, considering that the number of output tokens is significantly higher.

Converting energy consumption into energy costs achieved by multiplying the energy consumption by 0.04 Euros per kWh and to make a reliable value to compare with Online LLM APIs it will be multiplied by 4. The final values for showing costs per 1M token can be seen in Figure 21. Deepseek-r1:70b costs 2.69 euros per 1M tokens and llama3.3:70b costs 2.82 euros. Except deepseek-r1:32b which cost 1.07 euros, other models were under 1 Euro per 1M tokens. Middle-sized models cost around 0.70 euros per 1M token and smaller models cost around 0.11 to 0.25 euros per 1M token. The cheapest model was Gemma3:1b which cost 0.069 euros per 1M tokens.

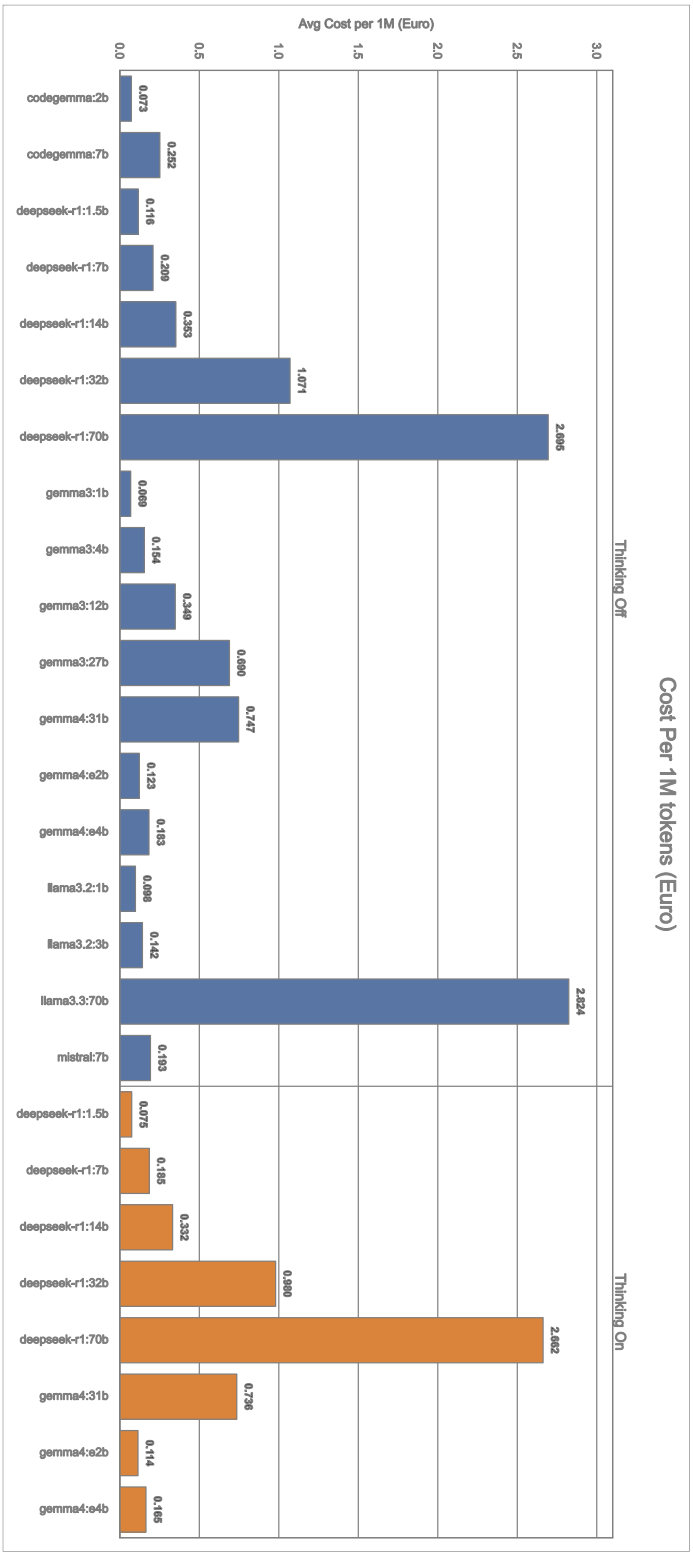

Figure 21 Energy Costs for 1M tokens

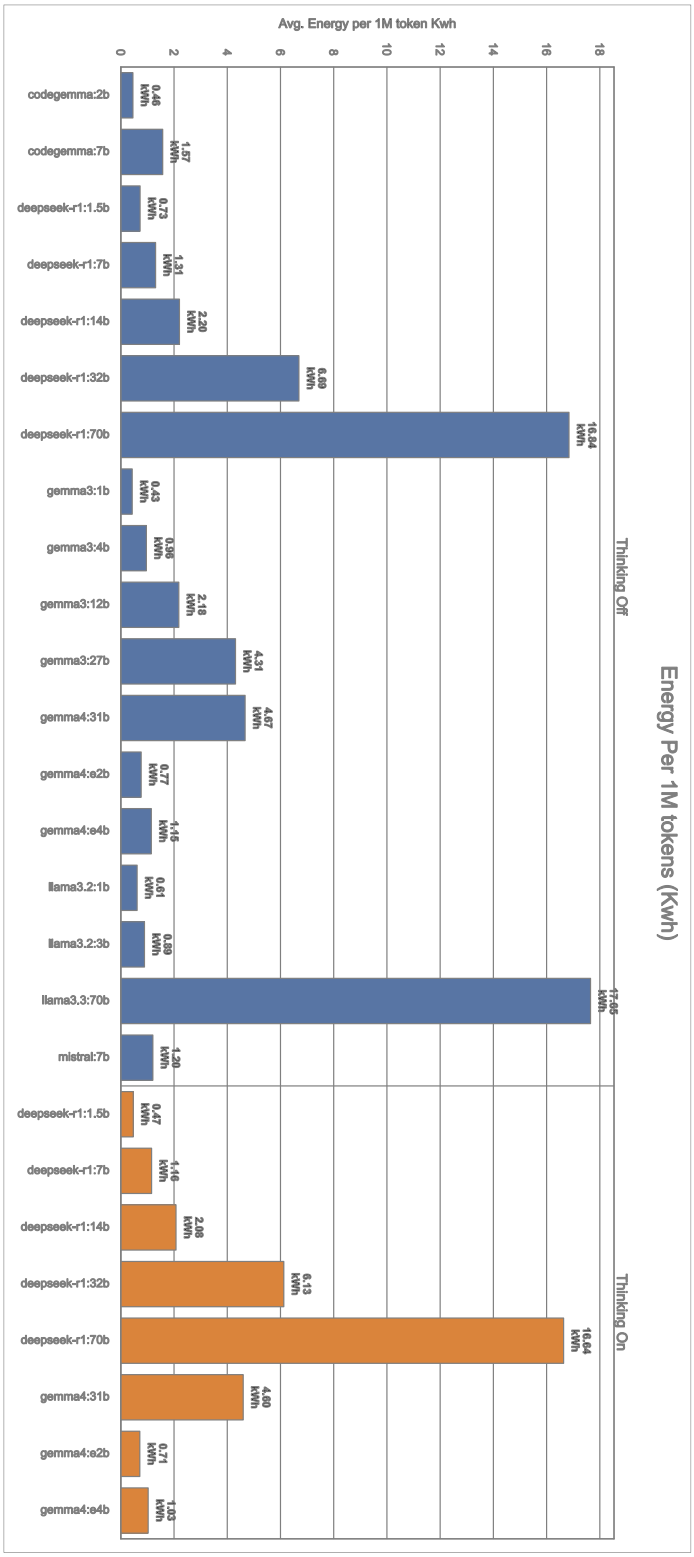

Figure 22 Average Energy Per 1M Token

Comparing local LLM cost with Online LLM API is done through the Table 7. Estimated values were obtained by multiplying the number of requests for each company per month by the cost of local or Online LLM cost per 1M. The number of requests per company per month was an estimation from Iternal.ai website (*AI Token Usage Guide (2026) — 10 Use Case Cost Profiles*, n.d.).

Table 7 comparing local vs online costs of LLM

Model	Local Cost per 1M token	Micro	Medium	Enterprises	Online API price per 1M token	Micro	Medium	Enterprises
codegemma:2b	0.07	1.46	18.23	218.71	0.6	12.00	150.00	1,800
codegemma:7b	0.25	5.04	62.98	755.76	0.6	12.00	150.00	1,800
deepseek-r1:1.5b	0.12	2.32	29.02	348.25	0.09	1.80	22.50	270
deepseek-r1:14b	0.35	7.05	88.15	1,057.77	0.14	2.80	35.00	420
deepseek-r1:32b	1.07	21.42	267.73	3,212.75	0.23	4.60	57.50	690
deepseek-r1:70b	2.69	53.89	673.68	8,084.20	1.72	34	430	5,160
deepseek-r1:7b	0.21	4.18	52.29	627.52	0.09	1.80	22.50	270
gemma3:12b	0.35	6.98	87.21	1,046.49	0.09	1.80	22.50	270
gemma3:1b	0.07	1.37	17.14	205.64	0.06	1.20	15.00	180
gemma3:27b	0.69	13.79	172.40	2,068.81	0.14	2.80	35.00	420
gemma3:4b	0.15	3.08	38.47	461.68	0.06	1.20	15.00	180
gemma4:31b	0.75	14.95	186.83	2,241.99	0.14	2.80	35.00	420
gemma4:e2b	0.12	2.45	30.64	367.72	0.03	0.60	7.50	90
gemma4:e4b	0.18	3.67	45.87	550.42	0.03	0.60	7.50	90
llama3.2:1b	0.10	1.97	24.56	294.75	0.01	0.20	2.50	30
llama3.2:3b	0.14	2.85	35.58	426.97	0.02	0.40	5.00	60
llama3.3:70b	2.82	56.48	706.04	8,472.47	0.34	6.80	85.00	1,020
mistral:7b	0.19	3.86	48.20	578.40	0.05	1.00	12.50	150

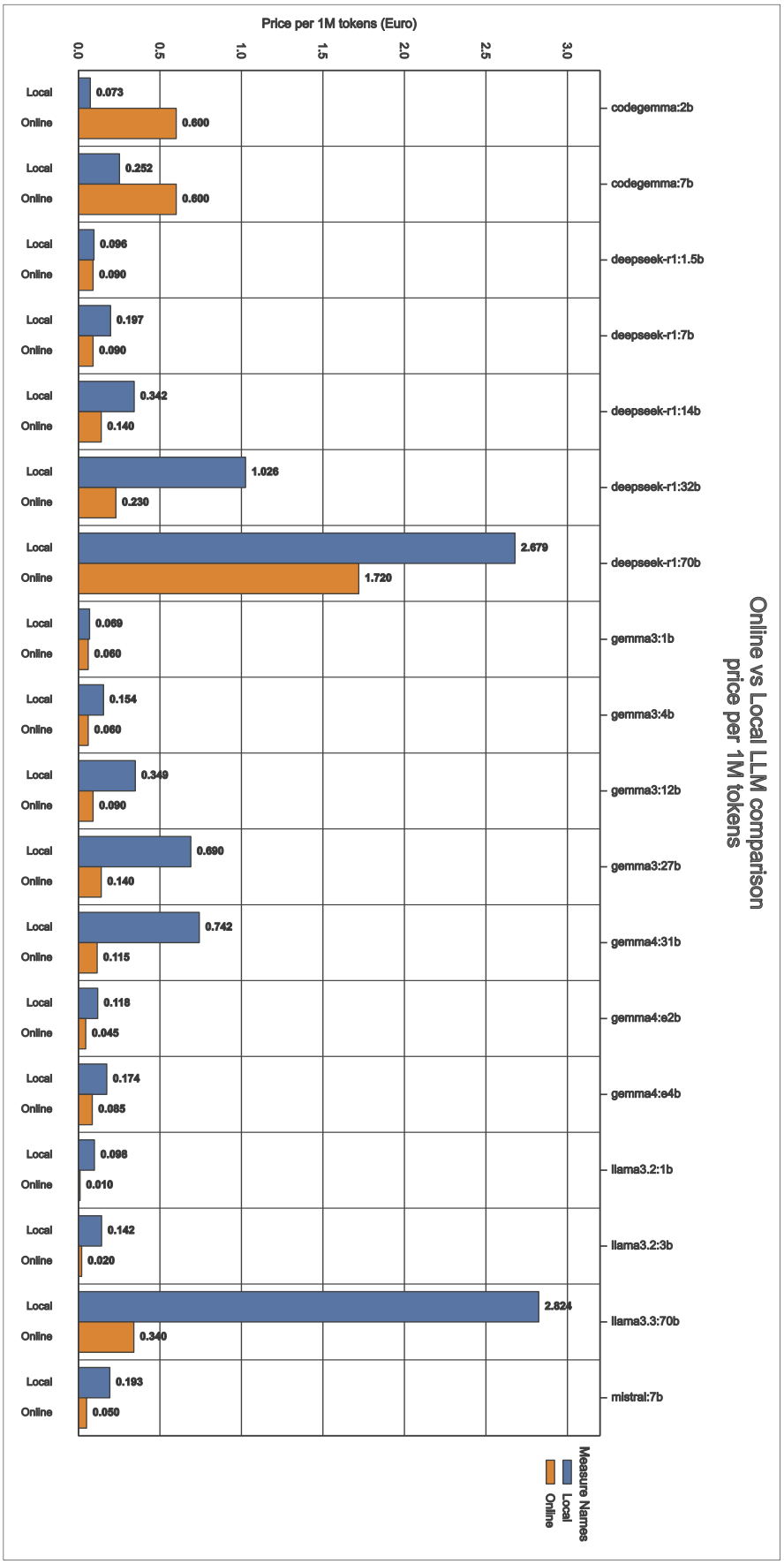

Figure 23 comparison of Online vs Local models

On Figure 23 a comparison of Local vs Online cost of models can be seen. This comparison shows that for almost all models it is better to use online API LLMs. One of the most striking conclusions that can be drawn from this comparison is the price of 70B parameter models. For high performance models, online API LLMs are significantly cost-effective. The most expensive local model is llama3.3:70b and costs 2.83 Euros per 1M tokens, while its online peer costs 0.34 Euros per 1M token and this is about 8 times cheaper than local deployment. Deepseek-r1:70b shows a similar trend, costing 2.68 Euros online and 0.11 Euros locally. Online models are generally cheaper than locally deployed models. The only exception is about codegemma models which are cheaper locally. The online costs for these models are 0.60 Euros per 1M token while for local deployment they cost 0.07 and 0.25 euros per 1M tokens for codegemma:2b and codegemma:7b respectively. This leads to conclusion that for open-source models with proper planning local deployment could be significantly cost effective.

4.2 Discussions

Multiple parameters affected the energy consumption of locally deployed large language models. The chain-of-thought reasoning capability is among these parameters. While Thinking capability was On, it significantly increased the count of output token, but it increased the duration and as a result the energy consumption of inference as well so the final energy per token remains almost equal to turning the Thinking Off. It is useful to have thinking capability ON because this produces more useful of answer to the query due to chain of taught reasoning.

In the order of output tokens, models are more different from each other and the hyper parameters of the experiment made differences. Thinking capability increases the number of output tokens by 4 times. This is an obvious result because Thinking capability before answering the question, returns their chain of thoughts inside a `<thinking>` tag and make their decisions based on the results of this tag. Therefore, number of output tokens should be considered in the measurement of energy consumption. Models generated 3 to 5 times more token when their thinking capability was On.

Keeping the models alive in memory and not dropping them after each inference seems to slightly reduce energy consumption.

There is a significant correlation between number of output tokens and total energy consumed. Figure 24 shows an instance, plotting number of output tokens and total energy consumed for Mistral:7b model.

Codegemma:2b generates significantly more tokens than other models. Not considering this model (because it has been finetuned for being a coding assistant), other models are almost in the same range.

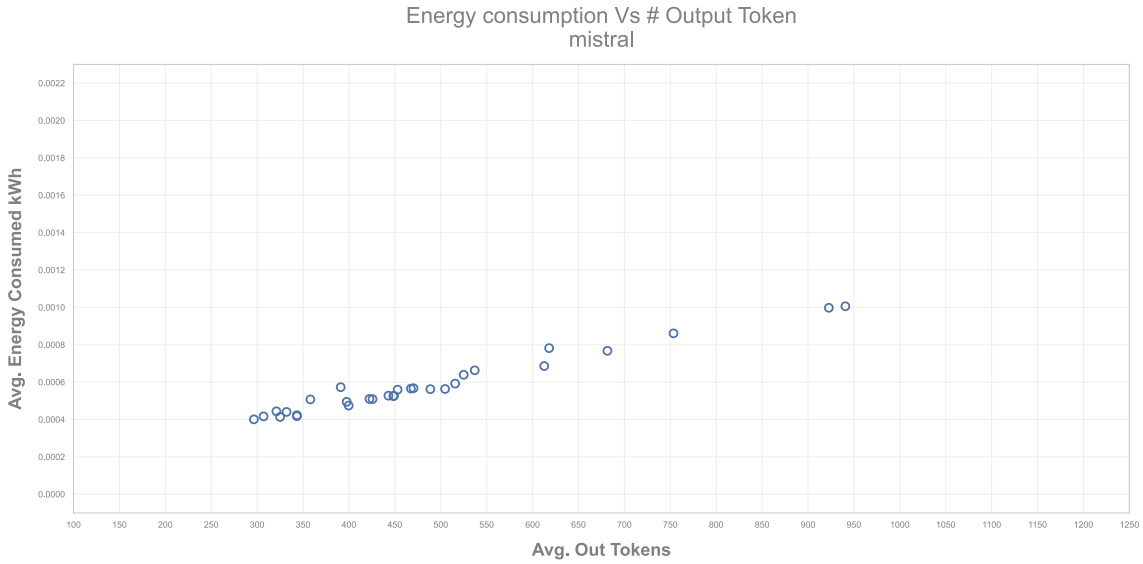

Figure 24 Number of Output token Vs Energy Consumption

Correlation of number of output tokens and total energy consumed per model can be seen on Table 8. Smallest value for correlation belongs to Llama3.2:1b (0.86).

Table 8 Correlation of number of output tokens and total energy consumed per model

Model Family	Model name	Correlation
CodeGemma	codegemma:2b	0.99
CodeGemma	codegemma:7b	0.93
Deepseek-r1	deepseek-r1:1.5b	0.91
Deepseek-r1	deepseek-r1:14b	0.99
Deepseek-r1	deepseek-r1:32b	0.97
Deepseek-r1	deepseek-r1:70b	0.99
Deepseek-r1	deepseek-r1:7b	0.99
Gemma3	gemma3:12b	0.99
Gemma3	gemma3:1b	0.99

Gemma3	gemma3:27b	0.99
Gemma3	gemma3:4b	0.98
Gemma4	gemma4:31b	0.99
Gemma4	gemma4:e2b	0.96
Gemma4	gemma4:e4b	0.98
Llama3.2	llama3.2:1b	0.86
Llama3.2	llama3.2:3b	0.97
Llama3.3	llama3.3:70b	0.97
Mistral	mistral:7b	0.98

Cost estimation for different companies based on their number of employees will be as Table 9 below.

Table 9 Monthly cost for energy consumption for different companies

Company	Average daily cost (Euro)	Monthly Cost (Euro)
micro	0.04	0.99
Small	0.27	5.91
Medium	1.12	24.63
Large	5.37	118.23
Enterprise	13.44	295.59

These numbers are only for LLM inference without considering any additional costs that AI product may have. These costs were mentioned in section 3.5.2 including training costs, infrastructure and computing costs, talent and compliance costs.

In case of large enterprises, a monthly bill of 300 Euros for running local LLMs is not considerable but adding the side cost of setup and maintenance of system can make it more complex for companies to have their own LLMs.

Regarding comparing the cost per 1M token from local to Online APIs it can be concluded that using online APIs is more convincing than running locally deployed LLMs. For almost all models (except codegemma) the total cost of using local LLM was higher than using online APIs. This estimated cost was using the multiplier to consider the additional costs of running LLM including infrastructure and computing costs where in online APIs it is combined with the final price.

Currently, many cloud-based LLM services are priced below the actual operating costs. Major providers of such services as OpenAI, Google, and Microsoft are in an early stage and at this phase gaining market share takes priority over profitability. As a result, businesses may deliver their services at an artificially low price that is not sustainable in the long run. Once market dominance is established, price increases are a logical and expected outcome. The present cost accounting of the LLM's models inference does not account for the training of newer models.

While long-term Power Purchase Agreements (PPAs) may allow large providers to reduce their energy costs and remain competitive at pricing their services, this advantage is not accessible to most businesses. This creates uncertainty and risk for organizations that rely entirely on cloud-based LLM services.

4.3 Conclusion

This section contained visualizations of the results of experiments. The experiments were testing 3 different benchmarks on the selected locally deployed LLMs and measured their number of tokens and energy consumption. The most important results were presented as different charts and a comparison between the results with another research presented to evaluate the results of the experiment. Finally, the results for the business cost estimation of using local LLMs and costs per 1M tokens were presented and conclusion was made based on these results in the final section of this chapter.

5 Conclusions

5.1 Summary of findings

This study measured how much electricity costs vary across different AI language models use when they run on local hardware. The study evaluated 18 models on three common business tasks: summarizing text (WikiHow), answering questions about financial reports (FinQA), and generating code (SWE-bench).

The findings are Summarized below

1. Number of output tokens is the main factor to estimate the energy consumption of the model. Almost all models showed a strong correlation between number of output tokens and energy consumption of the inference.
2. Thinking capability among family models like Deepseek-r1 and Gemma4 produces a chain-of-thought reasoning before the eventual answer. While having this feature on, produces three to five times more tokens and increases the total energy consumption but the energy per token remained the same. The ultimate finding is that thinking mode increases total energy consumption per query while keeping per-token constant.
3. The model size has a clear but non-linear relationship with energy per token. The 70b models, Deepseek-r1:70b and Llama3.3:70b, consumed approximately 1.7e-5 kWh per token, almost ten times more than small models in the 1-2 billion parameters range, which consumed around 4e-7 kWh per token.
4. Keeping model in memory between inferences (the keep-alive option) slightly reduced energy consumption by avoiding repeated model loading, while it had a negligible effect on energy per token efficiency.
5. The business cost analysis from the employee usage approach showed that daily electricity cost of running a locally deployed LLM is modest. Approximately €1 for micro, €25 for medium-sized, and €296 for enterprise companies per month. These values represent only inference electricity costs at the Finnish spot price of €0.04/kWh and not including infrastructure, maintenance, or talent costs.

6. When comparing local models to online AI services, on cost per million token basis, and applying a multiplier of four to local energy costs to consider the additional costs of running LLMs (including infrastructure, compliance, talent, and the data cost, as reported by ICONIQ 2026), online services proved more cost-effective for almost all of the tested models. This result suggests that for most organizations, the financial justification for local deployment lies primarily on price risk mitigation, data privacy, regulatory compliance, and long-term operational control rather than in electricity cost savings alone.

5.2 Recommendations

Based on the findings of this research, multiple recommendations can be made for businesses and organizations that want to use large language models on local deployment basis. Some of them are as follows

1. Model selection should be compatible with complexity of the task. Smaller models in the range of 7 to 14 billion parameters show a satisfactory performance inferencing simple and routine business tasks such as summarizing documents and emails while consuming less energy compared with other alternatives which own more parameters. Therefore, Organizations should avoid using large LLMs. This further optimizes the electricity savings
2. The chain-of-thought reasoning feature, known as thinking capability, should be used selectively. This feature increases the number of outputs taken by 3 to 5 times and consumes more energy as well, although this feature is made to solve complex problems and queries and will not be necessary for simple and routine daily tasks. For smaller tasks, disabling this feature is recommended.
3. Enabling keep alive option in Ollama is advised for the businesses with frequent usage of LLM. It reduces the overhead energy by avoiding reloading the model repeatedly.
4. Smaller organizations with lower usage volume and limited infrastructure are better to use online API services, which proved to be more cost-effective than local deployment for most models evaluated in this study. Local deployment is more suitable when data privacy requirements are critically important or when usage volumes are consistently high.
5. Organizations should consider energy consumption as a significant cost factor when adopting AI tools. This research showed that energy costs are real and measurable and

businesses should include electricity consumption in their cost of ownership calculations when evaluating local LLM deployment.

5.3 Further extension for future studies

While this study provides empirical insights into the energy consumption of locally deployed LLMs, several directions remain open for future research.

1. This study focused on a single configuration using NVIDIA L40S GPUs. Future research should extend the experiments to a wide range of hardware setups, for example consumer-grade GPUs, edge devices, and CPU only systems. This will provide a more comprehensive understanding of how hardware architecture affects the energy consumption of different deployments.
2. The evaluated models in this research ranged from 1.5 to 70 billion parameters. The rapid evolution of the field will introduce newer models which can be used for measuring energy consumption. Mixture-of-experts models and AI Agents are among the new and unevaluated systems that can be used for future energy measurement.
3. This study focuses on measuring energy consumption of inference stage only. A full life cycle of a LLM includes training, fine tuning and adaptation for specific tasks and measuring these stages will give a full picture of total environmental effects and financial impact of local LLM deployments.
4. The benchmark dataset used in this study, while representative of common business tasks, were limited to text summarization, financial analysis, and code generation. Expanding the benchmark to include additional task types such as multilingual processing, document classification, and generative multimedia content would increase the generalization of the findings.
5. Future studies could investigate the relationship between model accuracy and energy efficiency by enabling automated evaluation metrics along with energy measurement. This will consider the quality of the output as well as the energy consumption of the local deployment of LLMs.
6. The business cost estimation framework proposed in this study relied on generalization AI usage statistics achieved from existing survey data, which may not accurately reflect

the usage patterns of specific organizations or industries. Future research should aim to collect more accurate and realistic data directly from employees to find empirical AI usage patterns. This could include tracking the actual frequency, duration, and task type of AI interaction at an individual level, which would allow for more accurate estimation of token consumption and energy per token or energy per employee calculation.

7. A comprehensive cost-benefit analysis should be incorporated into future frameworks, alongside energy consumption, taking into account productivity, time savings, and error reduction that provided by AI for employees. Such analysis can offer organizations a complete and more reliable decision-making framework for strategic planning of deployment of local large language model.

5.4 Technical contributions

This research makes several contributions to the existing body of work on energy measurement of locally deployed large language models.

1. This study deployed a structured benchmark pipeline that combines three business related task types, including text summarization, numerical report analysis, and code generation, into a single evaluation framework. While previous studies just relied on generic academic benchmarks, this framework was designed to simulate real-world business tasks.
2. This study is among the first study to empirically measure and compare the energy consumption of locally deployed LLMs, using simulated real-world business tasks and hardware monitoring through CodeCarbon library and professional GPU infrastructure.
3. A business cost estimation framework was proposed that translates empirical energy consumption data into operational electricity costs for different company sizes. This framework estimates the electricity bills of real-world AI usage patterns for business decision-making.
4. A token-based cost comparison methodology was developed to allow direct comparison between locally deployed LLMs and Online LLM APIs. This framework accounts the full cost structure of AI products through an evidence-based cost multiplier calculated from industry reports.

References

Agrawal, Prof. P. (2025a). Running LLMs Locally on Consumer Devices. *International Journal for Research in Applied Science and Engineering Technology*, *13*(4), 5433–5441. https://doi.org/10.22214/ijraset.2025.69433

Agrawal, Prof. P. (2025b). Running LLMs Locally on Consumer Devices. *International Journal for Research in Applied Science and Engineering Technology*, *13*(4), 5433–5441. https://doi.org/10.22214/ijraset.2025.69433

AI Token Usage Guide (2026)—10 Use Case Cost Profiles. (n.d.). Retrieved April 23, 2026, from https://web.archive.org/web/20260422140822/https://iternal.ai/token-usage-guide

Alexander Bick, Adam Blandin, & David Deming. (2025, February 27). *The Impact of Generative AI on Work Productivity*. https://www.stlouisfed.org/on-the-economy/2025/feb/impact-generative-ai-work-productivity

Aminabadi, R. Y., Rajbhandari, S., Awan, A. A., Li, C., Li, D., Zheng, E., Ruwase, O., Smith, S., Zhang, M., Rasley, J., & He, Y. (2022). DeepSpeed- Inference: Enabling Efficient Inference of Transformer Models at Unprecedented Scale. *SC22: International Conference for High Performance Computing, Networking, Storage and Analysis*, 1–15. https://doi.org/10.1109/SC41404.2022.00051

Argerich, M., & Patiño-Martínez, M. (2024). Measuring and Improving the Energy Efficiency of Large Language Models Inference. *IEEE Access*, *PP*, 1–1. https://doi.org/10.1109/ACCESS.2024.3409745

Busch, K., & Leopold, H. (2024). *Towards a Benchmark for Large Language Models for Business Process Management Tasks* (arXiv:2410.03255; Version 2). arXiv. https://doi.org/10.48550/arXiv.2410.03255

Caravaca, F., Cuevas, Á., & Cuevas, R. (2025). *From Prompts to Power: Measuring the Energy Footprint of LLM Inference* (arXiv:2511.05597; Version 1). arXiv. https://doi.org/10.48550/arXiv.2511.05597

Chen, Z., Chen, W., Smiley, C., Shah, S., Borova, I., Langdon, D., Moussa, R., Beane, M., Huang, T.-H., Routledge, B., & Wang, W. Y. (2022). *FinQA: A Dataset of Numerical Reasoning over Financial Data* (arXiv:2109.00122). arXiv. https://doi.org/10.48550/arXiv.2109.00122

Curran, K., Curran, E., Killen, J., & Duffy, C. (2024). The role of generative AI in cyber security. *Metaverse*, *5*, 2796. https://doi.org/10.54517/m2796

Dan Schwarz, Lawrence Phillips, Daniel Hnyk, & Nikos Bosse. (2025, November 26). *OpenAI's API Profitability in 2024*. https://web.archive.org/web/20251126210214/https://future-search.ai/openai-api-profit/

Dauner, M., & Socher, G. (2025). Energy costs of communicating with AI. *Frontiers in Communication*, *10*. https://doi.org/10.3389/fcomm.2025.1572947

Face H. (2024). *transformers: State-of-the-art Machine Learning for JAX, PyTorch and TensorFlow* (Version 4.24.0) [Python; OS Independent]. https://github.com/huggingface/transformers

Fernandez, J., Na, C., Tiwari, V., Bisk, Y., Luccioni, S., & Strubell, E. (2025). *Energy Considerations of Large Language Model Inference and Efficiency Optimizations* (arXiv:2504.17674). arXiv. https://doi.org/10.48550/arXiv.2504.17674

Honkanen, M. (2022, September 5). *Improve your B2B segmentation with employee count data*. https://www.vainu.com/blog/employee-count-data/

Hou, X., Zhao, Y., Liu, Y., Yang, Z., Wang, K., Li, L., Luo, X., Lo, D., Grundy, J., & Wang, H. (2024). Large Language Models for Software Engineering: A Systematic Literature Review. *ACM Trans. Softw. Eng. Methodol.*, *33*(8), 220:1-220:79. https://doi.org/10.1145/3695988

Husom, E. J., Goknil, A., Astekin, M., Shar, L. K., KÃ¥sen, A., Sen, S., Mithassel, B. A., & Soylu, A. (2025). Sustainable LLM Inference for Edge AI: Evaluating Quantized LLMs for Energy Efficiency, Output Accuracy, and Inference Latency. *ACM Trans. Internet Things*, *6*(4), 28:1-28:35. https://doi.org/10.1145/3767742

Husom, E. J., Goknil, A., Shar, L. K., & Sen, S. (2026). *The Price of Prompting: Profiling Energy Use in Large Language Models Inference* (arXiv:2407.16893). arXiv. https://doi.org/10.48550/arXiv.2407.16893

ICONIQ. (2025). *The State of AI in 2025*. https://www.iconiq.com/growth/reports/2025-state-of-ai

ICONIQPartners, F. (2026). *State of AI: Bi-Annual Snapshot*.

Jegham, N., Abdelatti, M., Koh, C. Y., Elmoubarki, L., & Hendawi, A. (2025). *How Hungry is AI? Benchmarking Energy, Water, and Carbon Footprint of LLM Inference* (arXiv:2505.09598). arXiv. https://doi.org/10.48550/arXiv.2505.09598

Ji, Z., & Jiang, M. (2026). A systematic review of electricity demand for large language models: Evaluations, challenges, and solutions. *Renewable and Sustainable Energy Reviews*, *225*, 116159. https://doi.org/10.1016/j.rser.2025.116159

Jimenez, C. E., Yang, J., Wettig, A., Yao, S., Pei, K., Press, O., & Narasimhan, K. (2024). *SWE-bench: Can Language Models Resolve Real-World GitHub Issues?* (arXiv:2310.06770). arXiv. https://doi.org/10.48550/arXiv.2310.06770

Kaufmann, A. (2025). Understanding the Total Cost of Inferencing Large Language Models. *Dell Technologies*.

Khan, N., Khan, Z., Koubaa, A., Khan, K., & Salleh, R. (2024). Global insights and the impact of generative AI-ChatGPT on multidisciplinary: A systematic review and bibliometric analysis. *Connection Science*, *36*. https://doi.org/10.1080/09540091.2024.2353630

Koupaee, M., & Wang, W. Y. (2018). *WikiHow: A Large Scale Text Summarization Dataset* (arXiv:1810.09305). arXiv. https://doi.org/10.48550/arXiv.1810.09305

Kwon, W., Li, Z., Zhuang, S., Sheng, Y., Zheng, L., Yu, C. H., Gonzalez, J. E., Zhang, H., & Stoica, I. (2023). *Efficient Memory Management for Large Language Model Serving with PagedAttention* (arXiv:2309.06180). arXiv. https://doi.org/10.48550/arXiv.2309.06180

LLaMa.CPP. (2025). LLaMa.CPP. https://github.com/ggml-org/llama.cpp

Lotz, J. F., Lopes, A. V., Peitz, S., Setiawan, H., & Emili, L. (2025). *Beyond Text Compression: Evaluating Tokenizers Across Scales* (arXiv:2506.03101). arXiv. https://doi.org/10.48550/arXiv.2506.03101

Ma, W., Liu, S., Lin, Z., Wang, W., Hu, Q., Liu, Y., Zhang, C., Nie, L., Li, L., & Liu, Y. (2024). *LMs: Understanding Code Syntax and Semantics for Code Analysis* (arXiv:2305.12138). arXiv. https://doi.org/10.48550/arXiv.2305.12138

Manchanda, J., Boettcher, L., Westphalen, M., & Jasser, J. (2025). *The Open Source Advantage in Large Language Models (LLMs)* (arXiv:2412.12004; Version 3). arXiv. https://doi.org/10.48550/arXiv.2412.12004

Neal Vaidya, Fred Oh, & Nick Comly. (2023, October 19). *Optimizing Inference on Large Language Models with NVIDIA TensorRT-LLM, Now Publicly Available*. NVIDIA Technical Blog. https://developer.nvidia.com/blog/optimizing-inference-on-llms-with-tensorrt-llm-now-publicly-available/

Niu, C., Zhang, W., Li, J., Zhao, Y., Wang, T., Wang, X., & Chen, Y. (2025). *TokenPowerBench: Benchmarking the Power Consumption of LLM Inference* (arXiv:2512.03024). arXiv. https://doi.org/10.48550/arXiv.2512.03024

Niu, C., Zhang, W., Zhao, Y., & Chen, Y. (2025). Energy Efficient or Exhaustive? Benchmarking Power Consumption of LLM Inference Engines. *SIGENERGY Energy Inform. Rev.*, *5*(2), 56–62. https://doi.org/10.1145/3757892.3757900

Pan, G., Chodnekar, V., Roy, A., & Wang, H. (2025). *A Cost-Benefit Analysis of On-Premise Large Language Model Deployment: Breaking Even with Commercial LLM Services* (arXiv:2509.18101). arXiv. https://doi.org/10.48550/arXiv.2509.18101

Pronk, K., & Zhao, Q. (2025). *Benchmarking Energy Efficiency of Large Language Models Using vLLM* (arXiv:2509.08867; Version 1). arXiv. https://doi.org/10.48550/arXiv.2509.08867

Sachin Gopal Wani, Tanisha Khurana, David Ellison, Matthew Ziegler, & Jarrett Upton. (2025, May 23). *On-Premise vs Cloud: Generative AI Total Cost of Ownership (2025 Edition)*. Lenovo Press. https://lenovopress.lenovo.com/lp2225-on-premise-vs-cloud-generative-ai-total-cost-of-ownership-2025-edition

Samsi, S., Zhao, D., McDonald, J., Li, B., Michaleas, A., Jones, M., Bergeron, W., Kepner, J., Tiwari, D., & Gadepally, V. (2023). *From Words to Watts: Benchmarking the Energy Costs of Large Language Model Inference* (arXiv:2310.03003). arXiv. https://doi.org/10.48550/arXiv.2310.03003

Touvron, H., Lavril, T., Izacard, G., Martinet, X., Lachaux, M.-A., Lacroix, T., Rozière, B., Goyal, N., Hambro, E., Azhar, F., Rodriguez, A., Joulin, A., Grave, E., & Lample, G. (2023). *LLaMA: Open and Efficient Foundation Language Models* (arXiv:2302.13971). arXiv. https://doi.org/10.48550/arXiv.2302.13971

Wei, J., Wang, X., Schuurmans, D., Bosma, M., Ichter, B., Xia, F., Chi, E., Le, Q., & Zhou, D. (2023). *Chain-of-Thought Prompting Elicits Reasoning in Large Language Models* (arXiv:2201.11903). arXiv. https://doi.org/10.48550/arXiv.2201.11903

WSTAR – Wasa Zero Emission Data Centre. (n.d.). WSTAR: Wasa Zero Emission Data Centre. Retrieved April 13, 2026, from https://wstardatacenter.fi/

Yu, H., Yang, Z., Pelrine, K., Godbout, J. F., & Rabbany, R. (2023). *Open, Closed, or Small Language Models for Text Classification?* (arXiv:2308.10092). arXiv. https://doi.org/10.48550/arXiv.2308.10092

Zachary Skidmore. (2026, March 20). *Microsoft emissions up 23% since 2020, company blames AI data centers*. https://www.datacenterdynamics.com/en/news/microsoft-emissions-up-23-since-2020-blames-ai-data-centers/

Zhang, H., Yu, P. S., & Zhang, J. (2025). A Systematic Survey of Text Summarization: From Statistical Methods to Large Language Models. *ACM Comput. Surv.*, *57*(11), 277:1-277:41. https://doi.org/10.1145/3731445

Zhang, Z. (Jack), Shi, J., & Tang, S. (2025). *Cloud or On-Premise? A Strategic View of Large Language Model Deployment* (SSRN Scholarly Paper No. 5296479). Social Science Research Network. https://doi.org/10.2139/ssrn.5296479