**Measuring Prompt Energy in LLMs**

**Research Question**

How do measurable prompt characteristics—such as word count, syntactic complexity, sentiment, and frequency of specific semantic categories—influence the energy consumption of large language models (LLMs)?

**System Requirements**

* Collect a diverse set of prompts varying in length, complexity, sentiment, and topic.
* Measure or estimate the energy usage for each prompt processed by a large language model.
* Record prompt features: word count, syntactic complexity, sentiment, and frequency of semantic categories.
* Analyze how these features relate to total and relative energy consumption.
* Present results with graphs, charts, and tables.
* Save data and results in a structured format (e.g., JSON/CSV database).
* Ensure experiments are repeatable with the same inputs and configurations.
* Summarize which prompt characteristics increase or decrease energy use.

**Data Collection**

**Source:** LMSYS-Chat-1M — 1,000,000 real user conversations  
**URL:** <https://huggingface.co/datasets/lmsys/lmsys-chat-1m>  
**Sample:** 75,000 first-turn user prompts (random)  
**Target Dataset:** ~67,500 clean prompts after filtering

**Rationale:**  
• Real-world diversity (≈210K users)  
• Represents typical LLM usage patterns  
• Pre-cleaned (PII removed)  
• Average length: 69.5 tokens

**Logistics:** Hugging Face dataset, non-redistributable, research use only.  
**Note:** AI-generated draft; verified and refined methods implemented.

**Experimental Plan**

* Run a pilot test with 100 prompts to verify the complete data pipeline.
* Execute full-scale runs through an automated API data collection system.
* Confirm correct energy logging, API reliability, and efficiency.
* Scale to 1M total prompts after successful validation.
* Implement real-time logging and error-handling for long-running jobs.

**Measurement Overview**

* Measures token-based energy variation across models for comparative analysis.
* Tracks tokens, latency, throughput, and estimated energy consumption.
* Prompts are processed individually with paragraph-length completions.
* Dataset goal: ~1M prompts in ~300K conversations (adjustable).
* **Budget capped at ~$400 total (token cost).**
* Emphasis on technical precision, cost efficiency, and scalability.

**LLM API Plan**

**Objective:** Measure model-level energy efficiency and cost-performance tradeoffs.

**Models:**  
• **GPT-4o-mini (OpenAI)** — baseline model for large-scale runs.  
• **Llama 3.1 8B (Groq)** — high-speed, balanced-cost bulk model.  
• **Mistral Large (Mistral.ai)** — open-weight contrast model for smaller subsets.

**Execution:**

* Send each prompt sequentially via automated data pipeline.
* Log latency, token counts, cost, and energy estimate per call.
* Use temperature = 0.3 for deterministic completions.
* Run bulk workloads on GPT-4o-mini and Llama; reserve Mistral for high-fidelity subsets.

**Estimated Total Cost (1M Prompts):**

| **Model** | **Intended Share** | **Est. Cost** | **Notes** |
| --- | --- | --- | --- |
| GPT-4o-mini | 50% | $35–40 | baseline energy data |
| Llama 3.1 8B | 40% | $12–15 | Groq-optimized |
| Mistral Large | 10% | $60–70 | open-weight evaluation |
| **Total** | 100% | **$110–150 USD** | within $400 cap |

**Analysis Framework**

* Automated data pipeline (gather\_energy\_data.ipynb) with incremental batching.
* Real-time token, latency, and cost tracking.
* Statistical analysis and visualization (stats.ipynb).
* Dynamic comparison of energy efficiency and token-per-second throughput.
* Export in JSON and CSV formats with reproducible metadata.
* Integration with CodeCarbon for energy estimation and model-specific calibration.

**Technical Implementation**

* **Data Extraction:** clean.ipynb — filters, parses, and prepares LMSYS dataset.
* **Energy Measurement:** gather\_energy\_data.ipynb — handles API calls, retries, and logging.
* **Analysis:** stats.ipynb — creates performance charts and comparative plots.
* **Data Storage:** Structured JSON/JSONL with versioned metadata.
* **Reproducibility:** Git-controlled notebooks and pinned dependencies (requirements.txt).
* **Resumable Processing:** Automatic checkpointing for long sequences.

**Key Improvements**

* **Automated Pipeline:** Real-time progress display, robust error handling.
* **Energy Estimation:** Model-specific calibration and CodeCarbon integration.
* **Comprehensive Analysis:** Multi-model visualization and correlation analysis.
* **Cost Optimization:** Reduced temperature and adaptive token limits.
* **Research Metadata:** Model versioning, timestamps, latency logs.
* **Export Capabilities:** Ready-to-publish CSV, JSON, and charts.

**Expected Outcomes**

* Identify key prompt characteristics driving higher energy use.
* Provide a reproducible framework for measuring LLM energy consumption.
* Quantify cost-energy tradeoffs across different architectures.
* Develop interactive visualizations for research and policy insights.
* Produce publishable findings supporting efficient prompt and model design.

**References**

<https://arxiv.org/pdf/2407.16893>  
<https://chat.deepseek.com/a/chat/s/5c44573e-18ac-4e7d-a6ab-a275731811d3>

Things to look for correlation with:

* Response length vs energy
* Prompt length vs efficiency
* Model efficiency comparison
* Time-to-first-token analysis
* Question type energy cost
* Domain-specific energy use
* Code vs text energy difference
* Vocabulary complexity impact
* Sentence structure efficiency
* Conversation depth cost
* Readability score correlation
* Special content energy cost
* Named entity density
* Quality vs energy tradeoffs
* Time-of-day performance
* Tokenization efficiency
* Error pattern analysis
* Optimal response length
* Model-specific optimization
* Batch processing efficiency