The Language–Energy Divide: Measuring Energy Costs of Multilingual LLM Inference

Naihao Deng **Alissa Shen** **Yiming Feng** **Joan Nwatu** **Jae-Won Chung**
Mosharaf Chowdhury **Yulong Chen** **Rada Mihalcea**
University of Michigan University of Aberdeen University of Cambridge
{dnaihao, mihalcea}@umich.edu yulong.chen@abdn.ac.uk

Project page: https://lit.eecs.umich.edu/language-energy-divide/
Code: https://github.com/MichiganNLP/language-energy-divide
Data: https://huggingface.co/datasets/MichiganNLP/language-energy-divide

Abstract

Large language models (LLMs) are increasingly deployed in multilingual settings, yet the energy costs of serving these models across different languages remain poorly understood. We present a systematic study of inference energy consumption across languages with ML.Energy framework (Chung et al., 2026a). We find striking disparities: *energy consumption per output token varies by up to 8.3× across languages, while total energy for a fixed set of requests varies by up to 179× between the cheapest (English, 17.6 kJ) and the most expensive (Pashto, 3,147 kJ) languages*. Our analysis shows that this disparity is driven by two compounding factors: (1) higher *per-token energy costs* for languages using complex or rare scripts, and (2) more tokens generated for low-resource languages. Moreover, we find a double cost + performance penalty: languages with the highest energy footprints also tend to achieve the lowest task accuracy. We reveal that the energy divide persists across models, hardware, and tasks, suggesting a systemic energy inequity in multilingual LLM deployment. Finally, we recommend that the community treat energy as a first-class evaluation axis, extend reporting checklists and model cards to include it, and adopt deployment-side mitigations for better energy efficiency.

1 Introduction

The rapid adoption of large language models (LLMs) has raised urgent questions about their environmental impact and computational costs (Patterson et al., 2022; CBRE, 2023, 2024, 2025; McKinsey & Company, 2023, 2024; Sebastian Moss, 2024; International Energy Agency, 2025; Helen Kou, 2025). As the field gradually shifts from research prototypes to deployed infrastructure, the dominant evaluation paradigm that focuses on task performance may be insufficient in capturing what matters for real-world use.

Multilingual benchmarks (Liang et al., 2020; Bandarkar et al., 2024; Shi et al., 2023; Xuan et al., 2025) in particular focus almost exclusively on per-language task performance, while overlooking the energy cost. On the other hand, prior work has shown that low-resource languages require more tokens to express semantically equivalent content (Ahia et al., 2023; Petrov et al., 2023), suggesting potentially unequal computational and energy costs. However, existing analyses primarily rely on token-level proxies such as sequence length or token counts, while the actual hardware-level energy consumption remains underexplored.

In this work, we address the following research question: *Does the language of a query systematically affect the energy cost of LLM inference, and if so, by how much?* If answering a question in a low-resourced language incurs significantly more energy as compared to answering the same question in English, LLM providers can face economic pressure to deprioritize low-resource languages, deepening the digital divide. To answer this question, we adopt the ML.Energy benchmarking framework (Chung et al., 2026a) to measure the energy consumption of several models across all 122 languages of the Belebele reading comprehension dataset (Bandarkar et al., 2024), and the 8 languages from the translated GSM8K math reasoning dataset (Cobbe et al., 2021) and the LM-Arena open-ended chat (Chiang et al., 2024). Within each model and hardware configuration, all languages are evaluated under identical inference conditions.

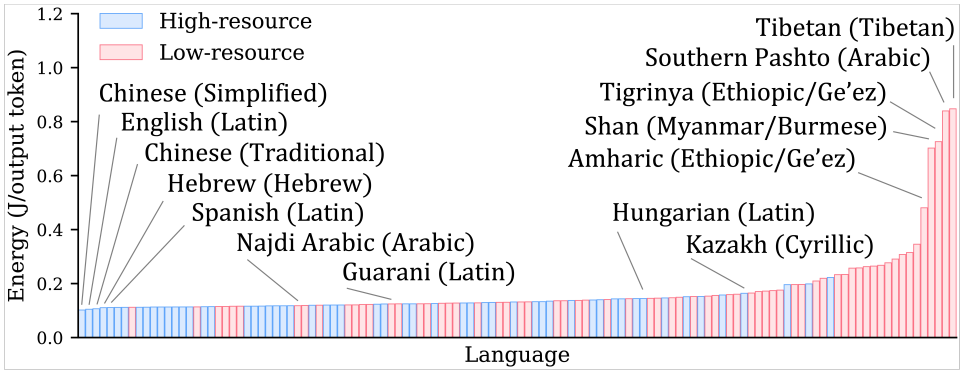
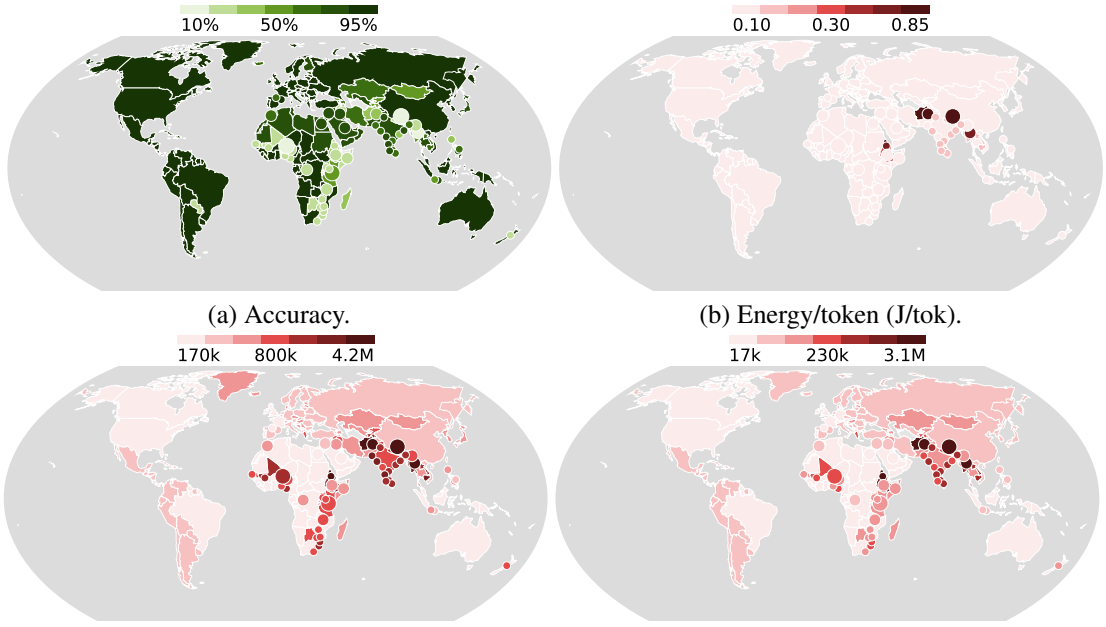


Figure 1: *Energy per output token* across 122 languages. Languages are sorted by energy cost and colored by resource level. The distribution is right-skewed, with the cluster of low-resource languages exceeding 0.4 J/token on the right.



(a) Accuracy.

(b) Energy/token (J/tok).

(c) Output token number.

(d) Total energy (J).

Figure 2: Geographic distribution of task accuracy and energy consumption (full results in Table 5. We note that the figure is intended as a qualitative geographic summary of our result.

We experiment across model families, GPUs, batch sizes, and tasks, and uncover several interconnected phenomena. First, *per-token energy* costs vary by up to 8.3× across languages, revealing substantial cross-lingual inequity in computational cost. Second, low-resource languages typically incur both the highest per-token energy costs and the most inflated token counts, yielding total energy costs up to 179× the total energy of English. At the same time, we also find that these languages simultaneously achieve low task accuracy, leading to a double cost + performance penalty. These disparities persist across all configurations we test, indicating a systemic energy inequity in multilingual LLM deployment. Figure 2 provides a qualitative geographic summary of our results. We close by arguing that current performance-centered evaluation conceals these disparities, and by laying out concrete recommendations for the community, including treating energy as a first-class evaluation axis, extending checklists and model cards to include energy, and adopting deployment-side mitigations for better energy efficiency.

2 Related Work

LLM energy and efficiency. A substantial body of work has studied the computational cost of training LLMs (Schwartz et al., 2020; Patterson et al., 2022), and more recently, the inference cost at scale (Desislavov et al., 2023; Chung et al.,

2026a). Efforts such as Zeus (You et al., 2023), CodeCarbon (Courty et al., 2024), and more recent inference-focused benchmarks (Niu et al., 2026) provide tools for measuring and tracking energy at the hardware level. Leveraging these tools, Chung et al. (2026a,b) measure the LLM energy consumption across a wide spectrum of tasks, and Luccioni et al. (2024) compare energy footprints across model families and modalities. However, prior work has largely focused on efficiency in the context of a single language (typically English) (Chung et al., 2026a), leaving multilingual inference costs unexplored. In this work, we leverage the existing tools to measure energy consumption in the multilingual setup, systematically measuring energy consumption across 122 languages.

Multilingual NLP and resource disparities. The gap in NLP performance between high- and low-resource languages is extensively documented (Conneau et al., 2018; Joshi et al., 2020; Liang et al., 2020; Hu et al., 2020; Shi et al., 2023; Chen et al., 2023; Yan et al., 2024; Xuan et al., 2025; Zan et al., 2026), and recent multilingual benchmarks have substantially broadened coverage (Bandarkar et al., 2024; Singh et al., 2024; Pomerenke et al., 2025; Romanou et al., 2025). In this paper, we leverage the multilingual benchmarks constructed by our community (Bandarkar et al., 2024) for the purpose of energy measurement. Recent work shows that tokenizers are themselves a source of cross-lingual inequity. Scripts with high character-to-token ratios pay more per query in API-priced systems (Rust et al., 2021; Ahia et al., 2023; Petrov et al., 2023), with up to 15× token inflation relative to English for some morphologically rich or non-Latin-script languages (Petrov et al., 2023). While model-quality and tokenizer disparities for low-resource languages are well-known, their compute and energy implications have not been studied at scale.

3 Preliminaries

Energy Consumption and Measurement. Every invocation of computation and memory movement on hardware consume both time and energy. Energy consumption is determined not only by *what* is computed, but also *how* it is computed on the hardware, requiring real measurement to derive insights. Similar to time, modern hardware expose power and/or energy consumption via a suite of hardware counters specific to each vendor, which can be sampled to measure the energy consumption of workloads. In this work, we use the Zeus library (You et al., 2023; ML.ENERGY, 2026) to measure and report the energy consumption of NVIDIA GPUs. Hardware counters are known to be accurate within a 5% error margin (NVIDIA, 2026) and have also been verified independently (Arafa et al., 2020).

Multilingual Dataset. We run our tests and analyses on multilingual datasets. Belebele (Bandarkar et al., 2024) is a multi-choice reading comprehension benchmark, where each item pairs a short passage with a four-option question, and a model must select the correct answer. Its items are parallel across all 122 languages (the same passages and questions, translated). We use all of the 900 questions per language from Belebele.

For analyses requiring task diversity, we also use GSM8K (Cobbe et al., 2021)), which is a benchmark of grade-school math word problems that require multi-step arithmetic reasoning; and LM-Arena (Chiang et al., 2024), which is a collection of real, open-ended user prompts to chat assistants with no reference answer. As neither of these latter datasets is natively multilingual, we machine-translate both so that all three tasks span the identical 8-language subset, including four high-resource languages of English (Latin), Chinese (Simplified), Russian (Cyrillic), French (Latin), and four low-resource languages of Southern Pashto (Arabic), Tigrinya (Ethiopic/Ge'ez), Shan (Myanmar/Burmese), Tibetan (Tibetan). We provide the details of how we process each dataset in Appendix B.

Language Resource Classification. We use the No Language Left Behind (NLLB) taxonomy (Costa-Jussà et al., 2022) to classify the resource level of the studied languages. Of the 122 Belebele languages, 56 are classified as high-resource and 66 as low-resource.

4 Results and Analysis

Common Setup. Each language is evaluated under identical conditions, served via a vLLM (Kwon et al., 2023) inference server on an HPC cluster. Following the ML.ENERGY Benchmark (Chung et al., 2026a), we identify a *steady-state* period defined as the window during which the server's batch size is saturated at its maximum configured value, which is the period most representative of

Res.	Lang.	J/tok	Tok/R	E_{tot} (J)	×En	Acc
High	English$_{(Latin)}$	0.10	186	17,588	1.0×	94.6
	Chinese$_{(Simplified)}$	0.10	403	36,983	2.1×	89.6
	French$_{(Latin)}$	0.11	288	29,420	1.7×	91.7
	Russian$_{(Cyrillic)}$	0.12	411	43,560	2.5×	90.9
Low	Tigrinya$_{(Ethiopic/Ge'ez)}$	0.73	3,695	2,413,328	137×	25.3
	Shan$_{(Myanmar/Burmese)}$	0.70	4,667	2,948,722	168×	10.6
	Tibetan$_{(Tibetan)}$	0.85	3,937	3,002,867	171×	21.9
	Southern Pashto$_{(Arabic)}$	0.84	4,164	3,146,541	179×	40.4

Table 1: Energy gap for Qwen3-8B on Belebele. "Res.", "Lang.", "J/tok", "Tok/R", "E_{tot} (J)", "×En", and "Acc" refer to resource level, language, energy per output token, output tokens per response, total energy in J, total energy relative to English, and accuracy, respectively.

long-term deployment conditions. Unless otherwise noted, all per-token energy values reported in this paper are steady-state values; this ensures that languages with short, "well-behaved" runs (e.g., English) are not penalized for cold-start overhead relative to languages whose runs are dominated by extended token generation.

4.1 Energy Varies Substantially Across Languages

Setup. We run our measurements using Qwen3-8B (Yang et al., 2025) on all 122 languages of the Belebele dataset (Bandarkar et al., 2024) described earlier. We construct a zero-shot chain-of-thought (CoT) (Wei et al., 2022) prompt for each language. Details of the construction pipeline and per-language curation are in Appendix B. The experiments are conducted on a single L40S GPU with the batch size set to 256.

Results. Figure 3 presents the energy per response. English uses 19.5 J/response, whereas Pashto consumes 3,496 J/response, Tibetan 3,337 J, Shan 3,276 J, and Tigrinya 2,681 J, presenting a 179× gap between the cheapest and most expensive languages. Generally, low-resource languages consume much more energy per request than high-resource languages such as English. There are notable exceptions, however: Egyptian Arabic (`arz_Arab`), a low-resource language, sits within the cheapest five in both the per-response and per-token views. The reason is because Egyptian Arabic shares its Arabic script with Modern Standard Arabic, a high-resource language with strong tokenizer coverage, and the cost benefit carries over to the low-resource variety. The same pattern holds for other low-resource Arabic varieties (Mesopotamian, Najdi, Moroccan, North Levantine), all of which fall in the cheap cluster.

The energy disparity is the *product* of two factors, the number of output tokens generated per request, and the energy consumed per token. Prior work has observed that low-resource languages elicit longer generations (Ahia et al., 2023; Petrov et al., 2023), and Table 1 reaffirms this that the four low-resource languages in our subset emit 20–25× more output tokens per response than English.

Beyond token count, however, our measurements reveal a second penalty: *the per-token energy also varies substantially across languages*, by up to 8.3×. Figure 1 plots per-token energy for all 122 languages in Belebele, ordered by value and colored by resource level. The distribution is heavily right-skewed: most languages cluster between 0.10 and 0.20 J/token, with a long tail of low-resource languages in non-Latin scripts (Tibetan, Pashto, Tigrinya, Shan, Amharic) exceeding 0.4 J/token.[1] The two penalties compound: the same five languages occupy the top of both Figure 1 and Figure 3, so a per-token cost that is ≈ 7× English combines with a token-count that is ≈ 20–25× English to yield a per-response cost that is ≈ 170–179× English. This compounding is not coincidental as these languages consume more memory per response (the amount of KV cache is proportional to the number of tokens in the sequence). Since serving systems are constrained by GPU memory, this forces servers serving such languages to run at smaller batch sizes that underutilize the GPU's compute units and therefore consume more energy per token (Chung et al., 2026b).[2]

Moreover, *low-resource languages are doubly penalized in performance and energy.* Figure 4 plots accuracy against per-token energy for all 122 languages. English achieves 94.6% accuracy at the lowest energy, while the highest-energy languages (Tibetan, Pashto, Tigrinya, Shan, Amharic) all fall at or below 43% accuracy, with Shan dropping to 10.6%. This is a *double cost + performance penalty, the model both fails to answer correctly and consumes far more energy on these languages*. These languages all use non-Latin scripts (Tibetan, Myanmar, Arabic, Ethiopic) with high character-to-token ratios. For these languages, the model emits long, repetitive output that combines elevated per-token cost with inflated token counts. We emphasize that

[1] The complete per-language values (energy per token, output tokens, total energy, and accuracy) for all 122 languages are provided in Appendix A.

[2] Smaller batch sizes underutilize the hardware's compute units and poorly amortize hardware overheads like static power draw, leading to higher energy consumption per FLOP.

this is a co-occurrence rather than a causal relationship: high per-token energy does not cause low accuracy, nor vice versa. Both are downstream symptoms of the same upstream factors, tokenizer under-coverage and sparse pretraining representation for these languages, which simultaneously inflate token counts, and thus energy, and degrade model quality.

Table 2 presents four representative generations alongside the English output for the same prompt. We identify four recurring phenomena:

Token explosion. For Shan, Pashto, Tibetan, Tigrinya, Burmese, and Amharic, the model emits up to $\sim$4,700 output tokens per question, entering a repetitive loop (Holtzman et al., 2020) unable to produce a short answer. This regime is the most energy-wasteful: both the per-token cost and the token count are elevated.

Over-generation. Latvian, Lithuanian, Tswana, and Wolof generate more tokens than English at a similar per-token cost, suggesting the model partially processes the language but responds at length, reflecting uncertainty or partial understanding.

High character-to-token ratio. Scripts such as Myanmar (Burmese, Shan), Tibetan, Ethiopic (Amharic, Tigrinya), and Indic scripts (Gujarati, Kannada, Odia, Malayalam) require multiple Unicode characters per syllable and are tokenized inefficiently by models trained primarily on Latin-script text (Ahia et al., 2023; Petrov et al., 2023). By contrast, Simplified Chinese (zho_Hans) matches English at $\approx$0.10 J/token despite its non-Latin script, because modern Chinese is tokenized efficiently.

Code-switching. For some low-resource languages the model responds in English or another high-resource language (Winata et al., 2021).

We note that the energy disparity reflects both modeling choices and intrinsic linguistic structure. On the modeling side, tokenizers trained on Latin-dominated corpora allocate few merges to underrepresented scripts (Petrov et al., 2023). On the linguistic side, scripts and morphologies differ in information density: abugidas like Ethiopic and Tibetan encode syllables across multiple Unicode codepoints that subword tokenizers fragment inefficiently (Ahia et al., 2023; Velayuthan and Sarveswaran, 2025); agglutinative morphology (Finnish, Turkish, Korean) packs root, tense, case, and agreement into single long words that BPE struggles to segment along morpheme boundaries (Rust et al., 2021; Toraman et al., 2023); and abjads like Arabic omit short vowels that the model must infer from context (Habash, 2010). Chinese is a counterexample: zho_Hans uses a non-Latin script but matches English at 0.10 J/token, showing that sufficient training-data coverage (Yang et al., 2025) can offset typological complexity.

4.2 Energy Cost Disparity Persists Across Models

Setup. We compare five models spanning three families and a range of sizes, Qwen3-8B, Qwen3-14B, and Qwen3-32B (Yang et al., 2025), gemma-3-27B (Team et al., 2025), and Llama-3.1-8B-Instruct (Grattafiori et al., 2024), on the 8-language subset specified in § 3. All models are served on L40S GPUs with the batch size set to 256.

Results. Table 3 reports per-token energy for all five models. *The cross-language disparity is present at every model size and in all three families.* The ratio between the most and least expensive of the 8 languages is $8.3\times$ for Qwen3-8B, $12.2\times$ for Qwen3-14B, $9.0\times$ for Qwen3-32B, $8.2\times$ for gemma-3-27B, and $7.1\times$ for Llama-3.1-8B-Instruct. English or Chinese is the cheapest language, and Tibetan or Shan is the most expensive on every model, and the high- and low-resource groups never overlap. The absolute per-token energy rises with model size (e.g. English from 0.10 J / token on Qwen3-8B to 0.17 on Qwen3-14B and 0.51 on Qwen3-32B). However, the divide does not diminish with scale. Across an 8B-to-32B span for Qwen 3 models, the four low-resource languages remain several times more expensive per token than the four high-resource ones at every size.

4.3 Energy Cost Disparity Persists Across Inference Configurations

Setup. We provide studies into two GPU types (NVIDIA L40S of 48G GPU memory and RTX 6000 Pro Blackwell of 96G GPU memory) and batch sizes ($\in \{16, 32, 64, 128, 256, 512\}$) for the inference configurations. We run Qwen3-8B and use the same 8-language subset (four high-resource and four low-resource) as § 4.2.

Results. In Figure 5, we observe that the *per-token energy differs across GPUs, but the energy disparity across languages persists at every batch size.* For instance, simplified Chinese remains one

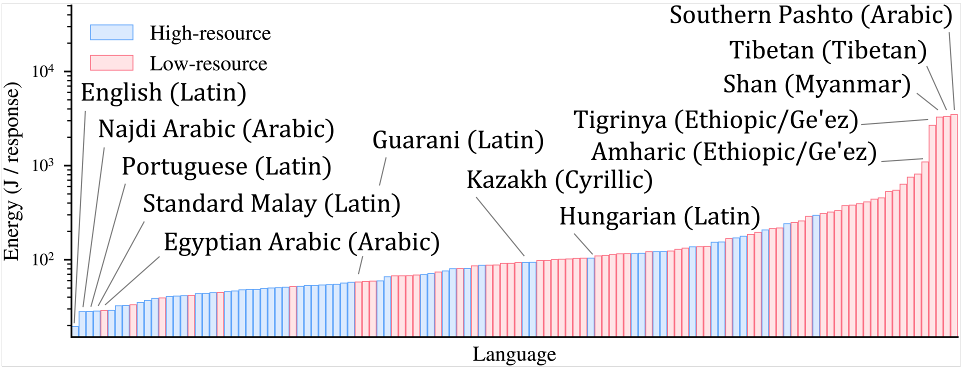


Figure 3: Energy per response across 122 languages (Qwen3-8B, zero-shot CoT, Belebele, log-scale). Per-response energy compounds the per-token cost (Figure 1) with output-token count, capturing the end-to-end energy a user pays per query. The disparity spans 19.5 J/response (English) to 3,496 J/response (Pashto), a 179× gap.

Input Lang.	Model Outputs	Comments
Tigrinya (Ethiopic/ Ge'ez)	…እዚ ምስ ከመሰረት ኣብ ዓወታት ምስ ተጠቒሱ ዘሎ ጽሑፍ ኣብ ውሽጢ እታ ቤተ-ክርስትያን ዝነበረ ፍርሒ ንምንካይ ከም ዝሓገዘ እተገልጸ እንታይ እዩ? (repeated 45 times)… *(translated) ...Based on this, in relation to victories/achievements, what is described in the mentioned text as having helped to reduce the fear that existed inside the church?...*	**Token explosion.** The sentence repeated 45 times.
Latvian (Latin)	Sāksim ar to, ka izlasīsim fragmentu, lai saprastu, ko tas saka par pareizo akordeona spēlēšanas padomiem… *(translated) Let's start by reading the excerpt to understand what it says about the correct tips for playing the accordion...*	**Over-generation.** There are 896 tokens generated, (3.4×English).
Tibetan (Tibetan)	ཁྱེད་ཀྱི་ཨང་གྲངས་སོགས་ནང་དུ་གཟིགས་ཕུར་བཞིན་ནི་གཟིགས་ཕུར་བཞིན་ནི་དེ་ནས་གྱུར་འགྱུར་བའི་དགེ་སྟོན་དེ་དགོ་གི་ཁྱོད་ཀྱི་ཨང་གྲངས་གཟིགས་ཕུར་བཞིན་ནི་ཨང་གྲངས་རེད།… *(translated) Looking at your numbers and so on, looking, looking, then the changing teachings — those are your numbers, looking, the numbers...*	**High chars/token.** 0.78 chars/tok v.s. 4.84 chars/tok for English.
Khmer (Khmer)	Okay, let's break this down step by step. The question is asking who leaked the Panama …	**Code-switching.** Answer in English.
English (Latin)	The passage provides tips for playing the accordion effectively. It emphasizes that: …Thus, the only inaccurate tip is Option 1.	**Consistent and correct.**

Table 2: Representative Qwen3-8B generations on Belebele. We present common phenomena of the model on response generation; for contrast, we also show high-quality English outputs on the same task.

of the cheapest languages at every batch size, while Shan is the most expensive at most batch sizes. The per-token energy disparity stems primarily from sequence-length effects. Low-resource languages produce longer token sequences, which inflate KV cache size and require attention and other computations over longer contexts, both during prefill (generating the first output token) and decoding, increasing the amount of computation and memory accesses. As batch size grows, memory pressure rises, and these effects compound across requests.

4.4 Energy Cost Disparity Persists Across Tasks

Setup. We now test how the task would influence the energy cost disparity. We use three tasks that span the distinct usage scenarios end users have with LLMs, and the corresponding datasets described in § 3: knowledge-grounded question answering (Belebele (Bandarkar et al., 2024)), structured multi-step reasoning (GSM8K (Cobbe et al., 2021)), and open-ended conversational assistance (LM-Arena (Chiang et al., 2024)). We hold the model (Qwen3-8B), hardware (a single L40S), and serving configuration (batch size of 256) fixed on the same 8-language subset as § 4.2.

Results. Table 4 reports energy per output token under all three tasks. *The cross-language disparity is present in every task.* The ratio between the most and least expensive of the 8 languages is 8.3× for Belebele (reading comprehension), 11.6× for

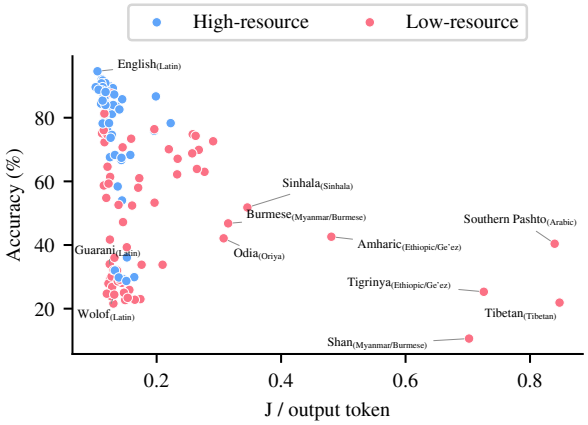


Figure 4: Output energy per token vs. accuracy across 122 languages (Qwen3-8B, zero-shot CoT, Belebele). The languages (typically the low-resource languages) on which the model has the highest energy per token are the ones on which it is least likely to answer correctly.

Res.	Lang.	Qwen3			Gemma3	Llama3.1
		8B	14B	32B	27B	8B
High	English(Latin)	0.10	0.17	0.51	0.59	0.09
	Chinese(Simplified)	0.10	0.17	0.56	0.67	0.10
	Russian(Cyrillic)	0.12	0.20	0.70	0.72	0.10
	French(Latin)	0.11	0.19	0.60	0.69	0.10
Low	Southern Pashto(Arabic)	0.84	1.36	1.00	0.88	0.44
	Tigrinya(Ethiopic/Ge'ez)	0.73	1.06	1.43	1.04	0.24
	Shan(Myanmar/Burmese)	0.70	1.64	2.83	4.79	0.66
	Tibetan(Tibetan)	0.85	2.02	4.60	1.63	0.52

Table 3: Per-language energy per output token (J / output token) across five models. The cross-language disparity reproduces on every model, and the high- and low-resource groups never overlap on any model.

GSM8K (math), and 10.6× for LM-Arena (chat). English or Chinese is the cheapest, and Tibetan is the most expensive language in all three tasks. The high- vs. low-resource gap persists across all tasks, with the energy cost falling disproportionately on the four low-resource, non-Latin-script languages.

5 Actionable Items and Recommendations

Treat energy as a first-class evaluation axis. We advocate for *reporting energy consumption as a standard companion to existing performance metrics across NLP research*. Our findings clearly indicate that performance metrics such as accuracy alone are an incomplete picture of what a system delivers in deployment. For instance, two systems that achieve comparable accuracy at substantially different energy costs are not equivalent, and our

Resource	Language	J / output token		
		Belebele (reading)	GSM8K (math)	LM-A (chat)
High	English(Latin)	0.105	0.086	0.097
	Chinese(Simplified)	0.102	0.087	0.098
	Russian(Cyrillic)	0.118	0.089	0.105
	French(Latin)	0.113	0.091	0.107
Low	Southern Pashto(Arabic)	0.840	0.117	0.957
	Tigrinya(Ethiopic/Ge'ez)	0.726	0.872	0.946
	Shan(Myanmar/Burmese)	0.702	0.516	0.593
	Tibetan(Tibetan)	0.847	0.990	1.023

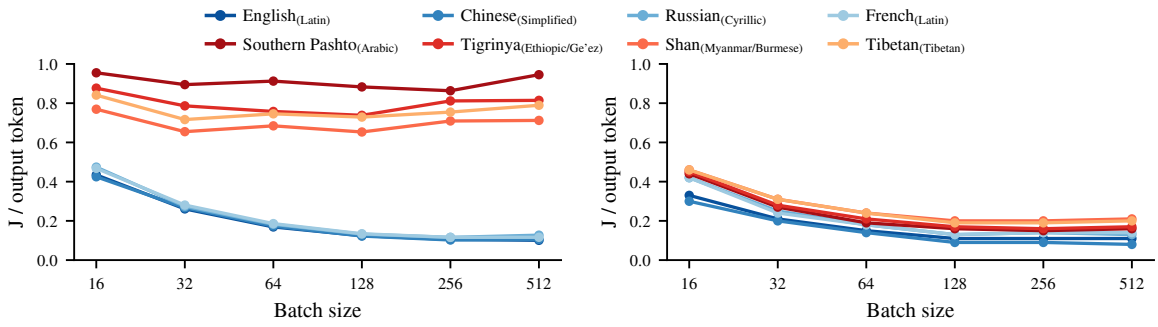
Table 4: Energy per output token for Qwen3-8B on the same 8-language subset across three datasets. The cross-language disparity persists under every task, with English or Chinese being cheapest and Tibetan most expensive in all three.

evaluation practices should reflect that.

We propose two concrete venues for energy reporting. First, extending the Responsible NLP Checklist (Rogers et al., 2021), to also include energy consumption as a standard checklist item, so that energy cost becomes visible alongside existing performance-centered metrics. This is by and large feasible, as energy is a measurable physical quantity that can be collected with existing tooling on the same hardware used for evaluation. Second, model developers should measure and report inference energy alongside the existing metrics in model cards. We note that training carbon emission is already an optional field of the model card spec (Luccioni et al., 2023; Faiz et al., 2024), but it obscures the amount of energy consumption by multiplying it with estimated carbon intensity; we believe the measurable physical quantity of energy will serve as a more concrete reporting metric.

Multilingual models should report energy consumption for each individual language. A model may incur unequal energy costs when serving different users, and these costs should be explicitly reported. A model that meets a 95% accuracy threshold on English at 0.10 J/token and 22% on Tibetan at 0.85 J/token represents, in deployment terms, two qualitatively different systems for two communities of users: cheap and reliable for one, expensive and broken for the other. Pooling these into a single "multilingual model" claim hides both the failure mode and the cost mode.

We argue that any "multilingual" coverage claims in models and papers should be backed by *per-language energy reports* in papers, model



(a) L40S.

(b) RTX 6000 Pro Blackwell.

Figure 5: Per token energy v.s. batch size of Qwen3-8B on L40S and RTX 6000 across 8 languages on Belebele.

cards, and on leaderboards. Further, he factors that drive disproportionate energy consumption in certain languages should be investigated (e.g., through ablation studies), rather than focusing solely on accuracy improvements. Re-framing evaluation in these terms changes which models are considered to have "succeeded" on multilingual coverage, and reorients the incentives that drive multilingual model development.

Language-aware serving optimization. Existing serving optimizations are unaware of the *language mixture* of workloads and choose configurations based on the *average* language of the expected incoming workload (Zhong et al., 2024; Agrawal et al., 2024; Zhu et al., 2025; Ma et al., 2025). However, as we have shown, language is a high-impact factor; it has broad serving implications, including the number of tokens, energy consumption, and so on. This means that when workloads are segmented into separate languages, each language may have different optimal configurations. For instance, different languages are read and spoken at different speeds (Brysbaert, 2019; Parachuk, 2022). In interactive applications like ChatGPT, languages that are read or spoken by native users at slower speeds (tokens per second) have longer time budgets to produce each output token, which allows the system to use larger batch sizes that consume less energy per token. Deeper and broader understanding of various languages (e.g., reading speed) will inform such optimizations and lead to higher efficiency.

Inference systems can also curb energy waste at serving time by adapting decoding constraints to the input. For multilingual deployments, we may use language detection to apply per-language maximum output lengths. We can leverage efforts from the generation community and apply adaptive output caps (Takase and Okazaki, 2019), early-stopping criteria (Wang et al., 2026), repetition penalties (Welleck et al., 2020), and so on, to ensure better output control.

Language-aware query routing. Rather than serving every input with a single generalist model, system designers can route queries based on the input language. One strategy is to route queries to smaller, specialized models when the input profile permits it (Ong et al., 2025; Chen et al., 2024). In the multilingual case, this means sending low-resource language queries to language-specialized models that are smaller and more efficient for their target language. Routing infrastructure already exists in production systems for cost reasons, and extending it to account for energy is a low-friction change with substantial aggregate impact.

A complementary strategy is to route eligible queries through a translate-then-process pipeline: translating non-English inputs to English, generating in English, and translating the output back to the source language. Because per-token energy costs in English are substantially lower and English generations require fewer tokens, this pipeline can reduce total energy consumption even after accounting for the overhead of two translation steps. Prior work has shown that such translate-then-process pipelines can also *improve* task accuracy for low-resource languages, since the core reasoning step occurs in the language where the model is good at (Shi et al., 2023; Chen et al., 2023; Kowtal et al., 2024; Ko et al., 2025). Energy efficiency and task performance, in these cases, can both be improved. We note however that this strategy may not be applicable in the case of tasks that hinge on cultural nuance, language-specific commonsense, idiomatic expression, or sociolinguistic register, where a translated output may be fluent

but not culturally faithful (Liu et al., 2025).

Treat per-language disparity as a deployment equity issue. The energy disparities translate directly into deployment-level inequities. Since energy costs are a primary driver of inference pricing, providers may, deliberately or not, impose higher costs on users of low-resource languages or exclude those languages from service to manage costs. Meanwhile, the aggregate environmental cost falls on all users, while the model failures fall disproportionately on speakers of underserved languages. We therefore believe that providers, alongside the research community, should treat per-language energy disparity as a deployment-time equity issue and to develop concrete mechanisms for addressing it, such as equity audits at model release, transparent per-language pricing, or subsidized inference for underserved languages.

6 Conclusion

We presented the first systematic measurement of inference energy consumption across languages in LLM serving, spanning 122 languages, alongside various models, inference configurations, and three tasks. Our main finding is a stark and persistent *language–energy divide*: per-token energy varies by up to $8.3\times$ across languages on a single model, and total energy per response varies by up to $179\times$ between English and Pashto. The divide persists across models, GPUs, batch sizes, and tasks, indicating a systemic property of multilingual LLM deployment. Worse, the languages most expensive to serve are also those on which models perform worst, imposing a double penalty on speakers of underserved languages. Our analysis revealed that the disparity is driven by two factors: low-resource and non-Latin-script languages incur both inflated per-token energy costs and longer output sequences. We argue that the community should treat energy as a first-class evaluation axis, report per-language energy in checklists and model cards, and adopt and innovate deployment-side mitigation methods. Closing the language–energy divide is not only an efficiency problem but an equity problem, and addressing it is essential if multilingual LLMs are to serve all their users fairly.

Acknowledgement

We thank the anonymous reviewers for their feedback on this work. This work was supported in part by NSF grants CCF-2450085 and CNS-2106184, DARPA ML2P Award HR0011-26-9-E190, and by grants from Ford, Laude Institute, OpenAI, and the Survival and Flourishing Fund. Jae-Won Chung was additionally supported by the Kwanjeong Educational Foundation and the Rackham Predoctoral Fellowship. Yulong Chen was supported by the DARPA program SciFy. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation, Ford, Laude Institute, Open AI or the Survival and Flourishing Fund.

Limitations

We discuss the limitations of our work here. First, due to resource limitations, our cross-hardware comparison is limited to NVIDIA L40S and RTX 6000 Pro Blackwell GPUs. We highlight that absolute per-token energy values are hardware-specific, and the magnitude of the disparity can differ across GPUs. Our findings should be re-validated as serving hardware evolves. Second, we do not evaluate commercial closed-source models such as GPT, Gemini, or Claude, which would require API-based energy estimation or access to provider infrastructure. Given that these models serve a substantial share of multilingual queries in deployment, extending our methodology to them is an important direction for future work, but it requires support from the companies. Third, Belebele and our translated GSM8K and LM-Arena prompts are standardized translations rather than natively authored text in each language. They may not fully reflect how each language is used in practice, particularly for orthography, register, and dialectal variation. The cross-language energy gap we measure is therefore best interpreted as a lower bound on the disparity faced by users producing natively authored, often noisier, real-world inputs.

Ethical Considerations

This work is fundamentally motivated by equity concerns, and we discuss them here.

Risk of language deprioritization. A central finding of this paper is that low-resource languages cost more energy to serve than high-resource ones. We are aware that publishing this result carries a dual-use risk: cost-conscious providers could use it to justify deprioritizing, rate-limiting, or excluding low-resource languages from service. We believe

the alternative, leaving these disparities unmeasured and undiscussed, is worse, because it allows the same outcomes to occur silently and without accountability. We have therefore framed our recommendations around *transparency* (per-language energy reporting), *mitigation* (language-aware serving, routing, and translate-then-process pipelines), and *equity audits at deployment*. We urge providers and researchers using our findings to do the same.

Whose costs, whose benefits. The energy costs we document are paid in aggregate, through electricity, water, and carbon, by all users and ultimately by the broader public, while the worst service quality is borne disproportionately by speakers of underserved languages. This asymmetry between who pays and who is served is itself an ethical issue, and one that performance-only evaluation has rendered invisible. We hope our work contributes to making it visible.

Environmental impact of this study. Our measurements required running LLM inference across many languages, models, hardware, and inference configurations, which itself consumed non-trivial energy. We view this as a one-time cost in service of changing community practice. To amortize it, we will release all per-language energy measurements, prompts, and processing scripts so that downstream researchers can build on our numbers rather than re-running the full sweep.

Translation quality and representation. Our translated prompts for GSM8K and LM-Arena, and the per-language instruction and primer for Belebele, were produced via NLLB-200 with back-translation quality control, and hand-curated where the automated pipeline failed. Despite these checks, machine-translated and even hand-curated prompts cannot fully substitute for native authorship by speakers of each language, and our results should not be taken as a verdict on the intrinsic difficulty or value of any language. They are a verdict on how current LLM stacks treat each language under current tokenization, training data, and serving choices.

Recommendations and downstream effects. We recommend, among other interventions, translate-then-process pipelines for tasks where the generation is largely language-agnostic. We note that uncritical application of such pipelines to culturally-loaded or sociolinguistically-rich tasks risks producing outputs that are fluent but culturally unfaithful, and we explicitly caution against this in § 5. Energy efficiency should not become a justification for erasing language-specific meaning.

References

Amey Agrawal, Nitin Kedia, Ashish Panwar, Jayashree Mohan, Nipun Kwatra, Bhargav Gulavani, Alexey Tumanov, and Ramachandran Ramjee. 2024. Taming throughput-latency tradeoff in LLM inference with Sarathi-Serve. In *OSDI*.

Orevaoghene Ahia, Sachin Kumar, Hila Gonen, Jungo Kasai, David Mortensen, Noah Smith, and Yulia Tsvetkov. 2023. Do all languages cost the same? tokenization in the era of commercial language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 9904–9923, Singapore. Association for Computational Linguistics.

Yehia Arafa, Ammar ElWazir, Abdelrahman ElKanishy, Youssef Aly, Ayatelrahman Elsayed, Abdel-Hameed Badawy, Gopinath Chennupati, Stephan Eidenbenz, and Nandakishore Santhi. 2020. Verified instruction-level energy consumption measurement for NVIDIA GPUs. In *Proceedings of the 17th ACM International Conference on Computing Frontiers*.

Howard Isaac Aronson. 1990. *Georgian: A reading grammar*. Slavica.

Lucas Bandarkar, Davis Liang, Benjamin Muller, Mikel Artetxe, Satya Narayan Shukla, Donald Husa, Naman Goyal, Abhinandan Krishnan, Luke Zettlemoyer, and Madian Khabsa. 2024. The belebele benchmark: a parallel reading comprehension dataset in 122 language variants. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 749–775, Bangkok, Thailand. Association for Computational Linguistics.

Winfried Boeder. 2005. The south caucasian languages. *Lingua*, 115(1-2):5–89.

Marc Brysbaert. 2019. How many words do we read per minute? a review and meta-analysis of reading rate. *Journal of memory and language*.

CBRE. 2023. Global data center trends 2023. https://www.cbre.com/insights/reports/global-data-center-trends-2023.

CBRE. 2024. Global data center trends 2024. https://www.cbre.com/insights/reports/global-data-center-trends-2024.

CBRE. 2025. Global data center trends 2025. https://www.cbre.com/insights/reports/global-data-center-trends-2025.

Lingjiao Chen, Matei Zaharia, and James Zou. 2024. FrugalGPT: How to use large language models while reducing cost and improving performance. *Transactions on Machine Learning Research*. Featured Certification.

Yulong Chen, Huajian Zhang, Yijie Zhou, Xuefeng Bai, Yueguan Wang, Ming Zhong, Jianhao Yan, Yafu Li, Judy Li, Xianchao Zhu, and Yue Zhang. 2023. Revisiting cross-lingual summarization: A corpus-based study and a new benchmark with improved annotation. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 9332–9351, Toronto, Canada. Association for Computational Linguistics.

Wei-Lin Chiang, Lianmin Zheng, Ying Sheng, Anastasios Nikolas Angelopoulos, Tianle Li, Dacheng Li, Hao Zhang, Banghua Zhu, Michael Jordan, Joseph E Gonzalez, and 1 others. 2024. Chatbot arena: An open platform for evaluating llms by human preference. *ArXiv preprint*, abs/2403.04132.

Jae-Won Chung, Jeff J. Ma, Ruofan Wu, Jiachen Liu, Oh Jun Kweon, Yuxuan Xia, Zhiyu Wu, and Mosharaf Chowdhury. 2026a. The ML.ENERGY benchmark: Toward automated inference energy measurement and optimization. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems Datasets and Benchmarks Track*.

Jae-Won Chung, Ruofan Wu, Jeff J. Ma, and Mosharaf Chowdhury. 2026b. Where do the joules go? diagnosing inference energy consumption. *ArXiv preprint*, abs/2601.22076.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. 2021. Training verifiers to solve math word problems. *ArXiv preprint*, abs/2110.14168.

Alexis Conneau, Ruty Rinott, Guillaume Lample, Adina Williams, Samuel Bowman, Holger Schwenk, and Veselin Stoyanov. 2018. XNLI: Evaluating cross-lingual sentence representations. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2475–2485, Brussels, Belgium. Association for Computational Linguistics.

Marta R Costa-Jussà, James Cross, Onur Çelebi, Maha Elbayad, Kenneth Heafield, Kevin Heffernan, Elahe Kalbassi, Janice Lam, Daniel Licht, Jean Maillard, and 1 others. 2022. No language left behind: Scaling human-centered machine translation. *ArXiv preprint*, abs/2207.04672.

Benoit Courty, Victor Schmidt, Sasha Luccioni, Goyal-Kamal, MarionCoutarel, Boris Feld, Jérémy Lecourt, LiamConnell, Amine Saboni, Inimaz, supatomic, Mathilde Léval, Luis Blanche, Alexis Cruveiller, ouminasara, Franklin Zhao, Aditya Joshi, Alexis Bogroff, Hugues de Lavoreille, and 11 others. 2024. mlco2/codecarbon: v2.4.1.

Radosvet Desislavov, Fernando Martínez-Plumed, and José Hernández-Orallo. 2023. Trends in ai inference energy consumption: Beyond the performance-vs-parameter laws of deep learning. *Sustainable Computing: Informatics and Systems*, 38:100857.

Ahmad Faiz, Sotaro Kaneda, Ruhan Wang, Rita Osi, Prateek Sharma, Fan Chen, and Lei Jiang. 2024. Llmcarbon: Modeling the end-to-end carbon footprint of large language models. In *International Conference on Learning Representations*, volume 2024, pages 24727–24741.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, and 1 others. 2024. The llama 3 herd of models. *ArXiv preprint*, abs/2407.21783.

Nizar Y Habash. 2010. *Introduction to Arabic natural language processing*. Morgan & Claypool Publishers.

Helen Kou. 2025. Power for ai: Easier said than built. `https://about.bnef.com/insights/commodities/power-for-ai-easier-said-than-built/`.

Brian George Hewitt. 1995. *Georgian: A structural reference grammar*, volume 2. John Benjamins Publishing.

Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. 2020. The curious case of neural text degeneration. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Junjie Hu, Sebastian Ruder, Aditya Siddhant, Graham Neubig, Orhan Firat, and Melvin Johnson. 2020. XTREME: A massively multilingual multi-task benchmark for evaluating cross-lingual generalisation. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, volume 119 of *Proceedings of Machine Learning Research*, pages 4411–4421. PMLR.

International Energy Agency. 2025. Electricity 2025, 2025.

Pratik Joshi, Sebastin Santy, Amar Budhiraja, Kalika Bali, and Monojit Choudhury. 2020. The state and fate of linguistic diversity and inclusion in the NLP world. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6282–6293, Online. Association for Computational Linguistics.

Hyunwoo Ko, Guijin Son, and Dasol Choi. 2025. Understand, solve and translate: Bridging the multilingual mathematical reasoning gap. In *Proceedings of the 5th Workshop on Multilingual Representation Learning (MRL 2025)*, pages 78–95, Suzhuo, China. Association for Computational Linguistics.

Nidhi Kowtal, Tejas Deshpande, and Raviraj Joshi. 2024. Chain-of-translation prompting (CoTR): A novel prompting technique for low resource languages. In *Proceedings of the 38th Pacific Asia Conference on Language, Information and Computation*, pages 645–655, Tokyo, Japan. Tokyo University of Foreign Studies.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. 2023. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the ACM SIGOPS 29th Symposium on Operating Systems Principles*.

Yaobo Liang, Nan Duan, Yeyun Gong, Ning Wu, Fenfei Guo, Weizhen Qi, Ming Gong, Linjun Shou, Daxin Jiang, Guihong Cao, Xiaodong Fan, Ruofei Zhang, Rahul Agrawal, Edward Cui, Sining Wei, Taroon Bharti, Ying Qiao, Jiun-Hung Chen, Winnie Wu, and 5 others. 2020. XGLUE: A new benchmark dataset for cross-lingual pre-training, understanding and generation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6008–6018, Online. Association for Computational Linguistics.

Chaoqun Liu, Wenxuan Zhang, Yiran Zhao, Anh Tuan Luu, and Lidong Bing. 2025. Is translation all you need? a study on solving multilingual tasks with large language models. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 9594–9614, Albuquerque, New Mexico. Association for Computational Linguistics.

Alexandra Sasha Luccioni, Sylvain Viguier, and Anne-Laure Ligozat. 2023. Estimating the carbon footprint of bloom, a 176b parameter language model. *Journal of machine learning research*, 24(253):1–15.

Sasha Luccioni, Yacine Jernite, and Emma Strubell. 2024. Power hungry processing: Watts driving the cost of ai deployment? In *Proceedings of the 2024 ACM conference on fairness, accountability, and transparency*, pages 85–99.

Jeff J. Ma, Jae-Won Chung, Jisang Ahn, Yizhuo Liang, Runyu Lu, Akshay Jajoo, Myungjin Lee, and Mosharaf Chowdhury. 2025. Cornfigurator: Automated planning for any-to-any multimodal model serving. *ArXiv preprint*, abs/2512.14098.

McKinsey & Company. 2023. Investing in the rising data center economy. https://www.mckinsey.com/industries/technology-media-and-telecommunications/our-insights/investing-in-the-rising-data-center-economy.

McKinsey & Company. 2024. How data centers and the energy sector can sate ai's hunger for power. https://www.mckinsey.com/industries/private-capital/our-insights/how-data-centers-and-the-energy-sector-can-sate-ais-hunger-for-power.

ML.ENERGY. 2026. Zeus. https://ml.energy/zeus.

Jihyung Moon, Hyunchang Cho, and Eunjeong L. Park. 2020. Revisiting round-trip translation for quality estimation. In *Proceedings of the 22nd Annual Conference of the European Association for Machine Translation*, pages 91–104, Lisboa, Portugal. European Association for Machine Translation.

Chenxu Niu, Wei Zhang, Jie Li, Yongjian Zhao, Tongyang Wang, Xi Wang, and Yong Chen. 2026. Tokenpowerbench: Benchmarking the power consumption of llm inference. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pages 32582–32590.

NVIDIA. 2026. NVIDIA Management Library (NVML). https://developer.nvidia.com/nvidia-management-library-nvml.

Isaac Ong, Amjad Almahairi, Vincent Wu, Wei-Lin Chiang, Tianhao Wu, Joseph E. Gonzalez, M Waleed Kadous, and Ion Stoica. 2025. RouteLLM: Learning to route LLMs from preference data. In *The Thirteenth International Conference on Learning Representations*.

Tara Parachuk. 2022. Speaking rates comparison table. https://www.voices.com/blog/languages-in-usa-speaking-rates-per-minute/#speaking-rates-comparison-table.

David Patterson, Joseph Gonzalez, Urs Hölzle, Quoc Le, Chen Liang, Lluis-Miquel Munguia, Daniel Rothchild, David R So, Maud Texier, and Jeff Dean. 2022. The carbon footprint of machine learning training will plateau, then shrink. *Computer*, 55(7):18–28.

Aleksandar Petrov, Emanuele La Malfa, Philip Torr, and Adel Bibi. 2023. Language model tokenizers introduce unfairness between languages. In *Thirty-seventh Conference on Neural Information Processing Systems*.

David Pomerenke, Jonas Nothnagel, and Simon Ostermann. 2025. The ai language proficiency monitor–tracking the progress of llms on multilingual benchmarks. *ArXiv preprint*, abs/2507.08538.

Reinhard Rapp. 2009. The backtranslation score: Automatic MT evalution at the sentence level without reference translations. In *Proceedings of the ACL-IJCNLP 2009 Conference Short Papers*, pages 133–136, Suntec, Singapore. Association for Computational Linguistics.

Ricardo Rei, Craig Stewart, Ana C Farinha, and Alon Lavie. 2020. COMET: A neural framework for MT evaluation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2685–2702, Online. Association for Computational Linguistics.

Anna Rogers, Timothy Baldwin, and Kobi Leins. 2021. 'just what do you think you're doing, dave?' a checklist for responsible data use in NLP. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 4821–4833, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Angelika Romanou, Negar Foroutan, Anna Sotnikova, Sree Harsha Nelaturu, Shivalika Singh, Rishabh Maheshwary, Micol Altomare, Zeming Chen, Mohamed A. Haggag, Snegha A, Alfonso Amayuelas, Azril Hafizi Amirudin, Danylo Boiko, Michael Chang, Jenny Chim, Gal Cohen, Aditya Kumar Dalmia, Abraham Diress, Sharad Duwal, and 38 others. 2025. INCLUDE: Evaluating multilingual language understanding with regional knowledge. In *The Thirteenth International Conference on Learning Representations*.

Phillip Rust, Jonas Pfeiffer, Ivan Vulić, Sebastian Ruder, and Iryna Gurevych. 2021. How good is your tokenizer? on the monolingual performance of multilingual language models. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3118–3135, Online. Association for Computational Linguistics.

Roy Schwartz, Jesse Dodge, Noah A Smith, and Oren Etzioni. 2020. Green ai. *Communications of the ACM*, 63(12):54–63.

Sebastian Moss. 2024. Meta's mark zuckerberg says energy constraints are holding back ai data center buildout, 2024.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Improving neural machine translation models with monolingual data. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 86–96, Berlin, Germany. Association for Computational Linguistics.

Freda Shi, Mirac Suzgun, Markus Freitag, Xuezhi Wang, Suraj Srivats, Soroush Vosoughi, Hyung Won Chung, Yi Tay, Sebastian Ruder, Denny Zhou, Dipanjan Das, and Jason Wei. 2023. Language models are multilingual chain-of-thought reasoners. In *The Eleventh International Conference on Learning Representations*.

Shivalika Singh, Freddie Vargus, Daniel D'souza, Börje F. Karlsson, Abinaya Mahendiran, Wei-Yin Ko, Herumb Shandilya, Jay Patel, Deividas Mataciunas, Laura O'Mahony, Mike Zhang, Ramith Hettiarachchi, Joseph Wilson, Marina Machado, Luisa Moura, Dominik Krzemiński, Hakimeh Fadaei, Irem Ergun, Ifeoma Okoh, and 14 others. 2024. Aya dataset: An open-access collection for multilingual instruction tuning. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 11521–11567, Bangkok, Thailand. Association for Computational Linguistics.

Sho Takase and Naoaki Okazaki. 2019. Positional encoding to control output sequence length. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 3999–4004, Minneapolis, Minnesota. Association for Computational Linguistics.

Gemma Team, Aishwarya Kamath, Johan Ferret, Shreya Pathak, Nino Vieillard, Ramona Merhej, Sarah Perrin, Tatiana Matejovicova, Alexandre Ramé, Morgane Rivière, Louis Rouillard, Thomas Mesnard, Geoffrey Cideron, Jean bastien Grill, Sabela Ramos, Edouard Yvinec, Michelle Casbon, Etienne Pot, Ivo Penchev, and 197 others. 2025. Gemma 3 technical report. *ArXiv preprint*, abs/2503.19786.

Cagri Toraman, Eyup Halit Yilmaz, Furkan Şahinuç, and Oguzhan Ozcelik. 2023. Impact of tokenization on language models: An analysis for turkish. *ACM Transactions on Asian and Low-Resource Language Information Processing*, 22(4):1–21.

Menan Velayuthan and Kengatharaiyer Sarveswaran. 2025. Egalitarian language representation in language models: It all begins with tokenizers. In *Proceedings of the 31st International Conference on Computational Linguistics*, pages 5987–5996, Abu Dhabi, UAE. Association for Computational Linguistics.

Junda Wang, Zhichao Yang, Dongxu Zhang, Sanjit Singh Batra, and Robert E Tillman. 2026. Estar: Early-stopping token-aware reasoning for efficient inference. *ArXiv preprint*, abs/2602.10004.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, brian ichter, Fei Xia, Ed H. Chi, Quoc V Le, and Denny Zhou. 2022. Chain of thought prompting elicits reasoning in large language models. In *Advances in Neural Information Processing Systems*.

Sean Welleck, Ilia Kulikov, Jaedeok Kim, Richard Yuanzhe Pang, and Kyunghyun Cho. 2020. Consistency of a recurrent language model with respect to incomplete decoding. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5553–5568, Online. Association for Computational Linguistics.

Genta Indra Winata, Samuel Cahyawijaya, Zihan Liu, Zhaojiang Lin, Andrea Madotto, and Pascale Fung. 2021. Are multilingual models effective in code-switching? In *Proceedings of the Fifth Workshop on Computational Approaches to Linguistic Code-Switching*, pages 142–153, Online. Association for Computational Linguistics.

Weihao Xuan, Rui Yang, Heli Qi, Qingcheng Zeng, Yunze Xiao, Aosong Feng, Dairui Liu, Yun Xing, Junjue Wang, Fan Gao, Jinghui Lu, Yuang Jiang, Huitao Li, Xin Li, Kunyu Yu, Ruihai Dong, Shangding Gu, Yuekang Li, Xiaofei Xie, and 13 others. 2025. MMLU-ProX: A multilingual benchmark for advanced large language model evaluation. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 1513–1532, Suzhou, China. Association for Computational Linguistics.

Jianhao Yan, Pingchuan Yan, Yulong Chen, Judy Li, Xianchao Zhu, and Yue Zhang. 2024. Gpt-4 vs. human translators: A comprehensive evaluation of translation quality across languages, domains, and expertise levels. *ArXiv preprint*, abs/2407.03658.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, and 1 others. 2025. Qwen3 technical report. *ArXiv preprint*, abs/2505.09388.

Jie You, Jae-Won Chung, and Mosharaf Chowdhury. 2023. Zeus: Understanding and optimizing {GPU} energy consumption of {DNN} training. In *20th USENIX Symposium on Networked Systems Design and Implementation (NSDI 23)*, pages 119–139.

Daoguang Zan, Zhirong Huang, Wei Liu, Hanwu Chen, Shulin Xin, Linhao Zhang, Qi Liu, Aoyan Li, Lu Chen, Xiaojian Zhong, Siyao Liu, Yongsheng Xiao, Liangqiang Chen, Yuyu Zhang, Jing Su, Tianyu Liu, RUI LONG, Ming Ding, and liang xiang. 2026. Multi-SWE-bench: A multilingual benchmark for issue resolving. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems Datasets and Benchmarks Track*.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q. Weinberger, and Yoav Artzi. 2020. Bertscore: Evaluating text generation with BERT. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Yinmin Zhong, Shengyu Liu, Junda Chen, Jianbo Hu, Yibo Zhu, Xuanzhe Liu, Xin Jin, and Hao Zhang. 2024. DistServe: Disaggregating prefill and decoding for goodput-optimized large language model serving. In *OSDI*.

Kan Zhu, Yufei Gao, Yilong Zhao, Liangyu Zhao, Gefei Zuo, Yile Gu, Dedong Xie, Tian Tang, Qinyu Xu, Zihao Ye, Keisuke Kamahori, Chien-Yu Lin, Ziren Wang, Stephanie Wang, Arvind Krishnamurthy, and Baris Kasikci. 2025. NanoFlow: Towards optimal large language model serving throughput. In *OSDI*.

Language	Code	Resource	J/tok	Output Tokens	Total Energy (J)	Accuracy
Southern Pashto	pbt_Arab	Low	0.84	3747646	3148022.6	40.4
Tibetan	bod_Tibt	Low	0.85	3543429	3011914.7	21.9
Shan	shn_Mymr	Low	0.70	4200313	2940219.1	10.6
Tigrinya	tir_Ethi	Low	0.73	3325414	2427552.2	25.3
Amharic	amh_Ethi	Low	0.48	2048079	983077.9	42.6
Sinhala	sin_Sinh	Low	0.35	2116100	740635.0	51.8
Burmese	mya_Mymr	Low	0.31	2154191	667799.2	46.8
Kannada	kan_Knda	Low	0.29	1963275	569349.8	72.6
Khmer	khm_Khmr	Low	0.28	1778891	498089.5	63.0
Gujarati	guj_Gujr	Low	0.27	1781503	481005.8	69.9
Eastern Panjabi	pan_Guru	Low	0.26	1556173	404605.0	74.3
Tamil	tam_Taml	Low	0.26	1540155	400440.3	68.8
Odia	ory_Orya	Low	0.31	1204527	373403.4	42.1
Telugu	tel_Telu	Low	0.26	1339006	348141.6	63.9
Lao	lao_Laoo	Low	0.23	1469386	337958.8	62.2
Assamese	asm_Beng	Low	0.23	1445450	332453.5	67.1
Malayalam	mal_Mlym	Low	0.26	1161153	301899.8	74.8
Sindhi	snd_Arab	Low	0.17	1678582	285358.9	61.0
Armenian	hye_Armn	Low	0.22	1267931	278944.8	70.1
Bengali	ben_Beng	High	0.22	1199146	263812.1	78.3
Nepali	npi_Deva	Low	0.21	1242288	260880.5	33.8
Swati	ssw_Latn	Low	0.16	1409901	225584.2	22.8
Maltese	mlt_Latn	High	0.14	1581018	221342.5	58.4
Igbo	ibo_Latn	Low	0.17	1286472	218700.2	23.0
Tajik	tgk_Cyrl	Low	0.17	1149221	195367.6	58.0
Fulfulde	fuv_Latn	Low	0.13	1475561	191822.9	21.6
Zulu	zul_Latn	High	0.15	1232890	184933.5	28.7
Marathi	mar_Deva	Low	0.20	896219	179243.8	76.4
Bambara	bam_Latn	Low	0.15	1119522	167928.3	22.7
Greek	ell_Grek	High	0.20	804328	160865.6	86.7
Central Kurdish	ckb_Arab	Low	0.18	859082	154634.8	33.8
Bengali (Latin)	ben_Latn	High	0.16	935815	149730.4	29.9
Hindi	hin_Deva	High	0.20	708854	141770.8	75.9
Tswana	tsn_Latn	High	0.15	940184	141027.6	29.2
Nyanja	nya_Latn	Low	0.15	846511	126976.7	25.0
Xhosa	xho_Latn	High	0.14	887370	124231.8	29.8
Jingpho	kac_Latn	Low	0.16	765203	122432.5	25.9
Nepali (Latin)	npi_Latn	Low	0.15	814114	122117.1	39.3
Wolof	wol_Latn	Low	0.13	880761	114498.9	24.4
Hausa	hau_Latn	Low	0.13	876624	113961.1	23.9
Modern Standard Arabic (Latin)	arb_Latn	High	0.15	720450	108067.5	36.1
Yoruba	yor_Latn	Low	0.15	713713	107057.0	23.4
Swahili	swh_Latn	High	0.14	760846	106518.4	54.0
Shona	sna_Latn	Low	0.14	756672	105934.1	28.6
Tsonga	tso_Latn	Low	0.14	741420	103798.8	29.1
Hindi (Latin)	hin_Latn	High	0.14	735045	102906.3	66.7
Maori	mri_Latn	Low	0.14	734697	102857.6	32.1

Continued on next page

Table 5 – *Continued from previous page*

Language	Code	Resource	J/tok	Output Tokens	Total Energy (J)	Accuracy
Southern Sotho	sot_Latn	High	0.13	785951	102173.6	32.0
Halh Mongolian	khk_Cyrl	Low	0.16	634984	101597.4	52.4
Kyrgyz	kir_Cyrl	Low	0.15	676822	101523.3	47.2
Northern Sotho	nso_Latn	Low	0.13	730114	94914.8	30.1
N. Azerbaijani	azj_Latn	Low	0.15	631317	94697.6	70.7
Ganda	lug_Latn	Low	0.13	726949	94503.4	26.8
Kinyarwanda	kin_Latn	Low	0.13	722516	93927.1	29.2
Urdu (Latin)	urd_Latn	Low	0.14	656631	91928.3	52.6
Hungarian	hun_Latn	High	0.14	646556	90517.8	85.8
Sinhala (Latin)	sin_Latn	Low	0.13	675122	87765.9	32.1
West Central Oromo	gaz_Latn	Low	0.14	615207	86129.0	28.1
Haitian Creole	hat_Latn	Low	0.13	657970	85536.1	61.4
Kazakh	kaz_Cyrl	High	0.16	534010	85441.6	68.3
Urdu	urd_Arab	Low	0.16	527972	84475.5	73.4
Thai	tha_Thai	High	0.14	603074	84430.4	82.6
Luo	luo_Latn	Low	0.12	687087	82450.4	24.7
Somali	som_Latn	Low	0.14	580127	81217.8	32.0
Georgian	kat_Geor	Low	0.20	400695	80139.0	53.3
Plateau Malagasy	plt_Latn	Low	0.13	586134	76197.4	36.0
Latvian	lvs_Latn	High	0.13	582382	75709.7	83.6
Czech	ces_Latn	High	0.13	557529	72478.8	89.3
Western Persian	pes_Arab	High	0.14	506687	70936.2	67.4
Ilocano	ilo_Latn	Low	0.12	580320	69638.4	41.7
Icelandic	isl_Latn	High	0.13	514387	66870.3	68.3
Moroccan Arabic	ary_Arab	Low	0.12	550284	66034.1	64.6
Northern Uzbek	uzn_Latn	High	0.13	495798	64453.7	73.7
Mesopotamian Arabic	acm_Arab	Low	0.12	532738	63928.6	72.3
Slovak	slk_Latn	High	0.12	519748	62369.8	78.3
Serbian	srp_Cyrl	Low	0.13	475857	61861.4	87.6
Sundanese	sun_Latn	Low	0.12	510165	61219.8	54.8
Tosk Albanian	als_Latn	High	0.13	462449	60118.4	81.2
Lingala	lin_Latn	Low	0.12	495406	59448.7	27.9
Kabuverdianu	kea_Latn	Low	0.12	463288	55594.6	58.7
Cebuano	ceb_Latn	Low	0.12	437536	52504.3	74.7
Estonian	est_Latn	High	0.12	430745	51689.4	76.2
Korean	kor_Hang	High	0.11	468108	51491.9	87.8
Afrikaans	afr_Latn	High	0.12	424235	50908.2	87.2
Macedonian	mkd_Cyrl	High	0.13	390445	50757.9	84.0
Guarani	grn_Latn	Low	0.12	418942	50273.0	34.0
Finnish	fin_Latn	High	0.12	397741	47728.9	78.4
Japanese	jpn_Jpan	High	0.11	432197	47541.7	78.2
Danish	dan_Latn	High	0.11	427814	47059.5	89.0
Romanian	ron_Latn	High	0.12	386978	46437.4	89.4
Ukrainian	ukr_Cyrl	High	0.13	356123	46296.0	87.3
Lithuanian	lit_Latn	High	0.12	384397	46127.6	83.9
Hebrew	heb_Hebr	High	0.11	414788	45626.7	84.4

Continued on next page

Table 5 – *Continued from previous page*

Language	**Code**	**Resource**	**J/tok**	**Output Tokens**	Total Energy (J)	**Accuracy**
Javanese	jav_Latn	Low	0.11	406097	44670.7	76.1
Russian	rus_Cyrl	High	0.12	370275	44433.0	90.9
Dutch	nld_Latn	High	0.11	397947	43774.2	88.1
Bulgarian	bul_Cyrl	High	0.12	362387	43486.4	89.9
Croatian	hrv_Latn	High	0.12	360991	43318.9	84.3
Slovenian	slv_Latn	High	0.12	360206	43224.7	87.2
Traditional Chinese	zho_Hant	High	0.11	392790	43206.9	88.8
Italian	ita_Latn	High	0.11	366630	40329.3	90.2
Polish	pol_Latn	High	0.12	333593	40031.2	84.0
Vietnamese	vie_Latn	High	0.11	359211	39513.2	88.6
Norwegian Bokmal	nob_Latn	Low	0.11	355727	39130.0	90.1
Turkish	tur_Latn	High	0.11	349338	38427.2	85.4
Waray	war_Latn	Low	0.12	306588	36790.6	59.3
Swedish	swe_Latn	High	0.11	330028	36303.1	89.6
Simplified Chinese	zho_Hans	High	0.10	362454	36245.4	89.6
Spanish	spa_Latn	High	0.11	329321	36225.3	90.8
North Levantine Arabic	apc_Arab	Low	0.11	315102	34661.2	75.1
Catalan	cat_Latn	High	0.11	306329	33696.2	89.7
German	deu_Latn	High	0.11	297520	32727.2	92.1
Najdi Arabic	ars_Arab	Low	0.12	253562	30427.4	75.2
Basque	eus_Latn	High	0.12	253348	30401.8	67.6
Tagalog	tgl_Latn	High	0.13	227736	29605.7	74.6
French	fra_Latn	High	0.11	259581	28553.9	91.7
Egyptian Arabic	arz_Arab	Low	0.12	224620	26954.4	81.3
Indonesian	ind_Latn	High	0.12	223840	26860.8	85.0
Modern Standard Arabic	arb_Arab	High	0.12	217531	26103.7	80.8
Standard Malay	zsm_Latn	High	0.12	217255	26070.6	88.1
Portuguese	por_Latn	High	0.11	223499	24584.9	90.2
English	eng_Latn	High	0.10	167621	16762.1	94.6

Table 5: Full results for all 122 languages by Qwen3-8B on Belebele. J/tok reports the energy per output token. Output Tokens reports the total number of tokens generated across 900 responses.

A Language List and Energy Statistics

Table 5 provides the complete list of all 122 Belebele languages with their resource level, energy per output token, total output tokens, total energy consumption, and task accuracy on Qwen3-8B under zero-shot CoT prompting.

A.1 Discussion

Table 5 surfaces several patterns that we briefly highlight here.

Apart from the analysis we conduct in § 4.1 that there is significant energy gap across languages, the languages with the highest output-token counts also exhibit the highest J/tok, and energy cost and accuracy are inversely related, we also note several other observations here.

Several languages labeled high-resource (Zulu 28.7, Tswana 29.2, Xhosa 29.8, Bengali-Latin 29.9) underperform languages labeled low-resource (Marathi 76.4, Javanese 76.1, Kabuverdianu 58.7). Therefore, *the High/Low resource binary is a coarse predictor.* Script familiarity and overlap with the pretraining mixture are plausible explanations for the variance in accuracy on this benchmark.

Latin transliterations reduce token counts substantially, but their effect on accuracy is inconsistent. Hindi-Latin loses only modestly relative to Devanagari (75.9 → 66.7), whereas Bengali-Latin collapses from 78.3 to 29.9 and Sinhala-Latin from 51.8 to 32.1. Romanization helps the tokenizer but can disrupt the model's lexical and morphological knowledge of the language. Therefore, *romanization is not a uniform win.*

There are outliers to our analysis. Georgian (`kat_Geor`) exhibits a relatively high J/tok (0.20) at a moderate output length ($\sim 4.0 \times 10^5$ tokens) and lands at 53.3% accuracy, which is an unusual position relative to other non-Latin scripts. This likely reflects intrinsic properties of the language, including its agglutinative morphology with up to eight verbal morphemes and polypersonal agreement (Aronson, 1990; Hewitt, 1995), and the relative isolation of the Kartvelian family and its unique Mkhedruli script, which shares no genetic relationship with other major writing systems (Boeder, 2005).

Figure 2 plots the geographic distribution of the results from Table 5. Note that the country-fill assignment necessarily simplifies multilingual states (e.g. Belgium is shown by Dutch, Switzerland by German, Mali by Bambara rather than the colonial language), and pin positions are illustrative rather than cartographically precise. However, we can still observe that Europe, the Americas, East Asia, and the anglophone parts of Africa and Oceania appear uniformly dark-green and pale-red (high accuracy while low energy cost), while the band running from West Africa through the Sahel, the Horn of Africa, the Himalayan plateau, and mainland Southeast Asia appears pale-green and dark-red (low accuracy while high energy cost).

B Experimental Setups

Belebele zero-shot CoT prompt construction. For Belebele (Bandarkar et al., 2024), we use a zero-shot chain-of-thought prompt covering all 122 languages of the Belebele test split. The prompt format, identical across languages up to translation, is:

> *<per-language instruction, ending with the final line "#### N">*
> *<P-label>: <passage>*
> *<Q-label>: <question>*
> *<O-label>: 1. <option 1> 2. <option 2> 3. <option 3> 4. <option 4>*
> *<per-language primer "Let's think step by step.">*

The per-language instruction is a translation of the canonical English instruction *"Read the passage and answer the multiple-choice question. Reason step by step. Then, on a new last line, write the number of the correct option (1, 2, 3, or 4) in exactly this format:"* followed by `#### N` on its own final line, which is used in every language as the parser anchor. The primer is a translation of *"Let's think step by step."*. The passage, question, and four answer options are taken directly from the Belebele test split for the target language. In addition, the Passage/Question/Options labels are all translated to the target language.

Per-language translation pipeline. We translate the instruction and primer for each Belebele language using NLLB-200 (Costa-Jussà et al., 2022), then back-translate (Rapp, 2009; Sennrich et al., 2016; Moon et al., 2020) every candidate and filter by BERTScore (Zhang et al., 2020) ≥ 0.85 and COMET (Rei et al., 2020) ≥ 0.75. This automated pipeline yields translations for 93 of the 122 Belebele languages. Of the remaining 29 languages, 1 is the source language (English) and requires no translation. For the remaining 28 languages, we hand-curate the instruction and primer using

Claude-4.7-Opus[3] and manually verify each back-translation. These 28 cases comprise 11 languages with dropped clauses (e.g. Bulgarian, French, Russian, Ukrainian), 10 with mistranslations (e.g. Korean, Bambara, Igbo), 1 partial (Fulfulde), , and 6 romanized Latin-script Belebele variants for which NLLB has no direct code (`arb_Latn`, `ben_Latn`, `hin_Latn`, `npi_Latn`, `sin_Latn`, `urd_Latn`). In all manual cases, we translate only the prose and keep the parser anchor `#### N` as the final line.

Translated GSM8K and LM-Arena. Similarly, the translated GSM8K and LM-Arena prompts in § 4.4 are produced with an NLLB-200 translation and back-translation quality-control pipeline (BERTScore and COMET), yielding 272 and 266 prompts that pass the quality bar in all 8 languages. We translate the original GSM8K (Cobbe et al., 2021) rather than directly use MGSM (Shi et al., 2023) so the math task spans the identical 8-language subset. All three tasks in Table 4 share the zero-shot prompt design: GSM8K appends the per-language "Let's think step by step." primer to the translated question (similar to the zero-shot CoT for Belebele), while LM-Arena uses the translated user message directly (the standard open-chat). To match the request count of the 900-request Belebele runs we issue each prompt three times, giving $272 \times 3 = 816$ and $266 \times 3 = 798$ effective requests, respectively. Because per-token energy is invariant to the total request count by construction, the differing effective request counts across tasks (900, 816, and 798) do not affect comparability.

Serving configuration and parallelism. All models are served via vLLM (Kwon et al., 2023), running inside a Singularity container, with the batch size (the maximum number of concurrent sequences) set to 256 unless otherwise noted. For the cross-model experiment (§ 4.2) on L40S GPUs, we run Qwen3-8B, Qwen3-14B, and Llama-3.1-8B-Instruct on a single GPU, Qwen3-32B and gemma-3-27b-it by pipeline parallelism with degree 2. For a batch size of 512, we use a 1800-request run (two passes over the 900-prompt set). Specifically, the 900-request run's steady-state window collapses because $900 < 2 \times 512$, whereas $1800 > 2 \times 512$ yields a valid window. Per-token energy is request-count-invariant by construction in the saturated-serving regime (Chung et al., 2026a), so this does not affect comparability across batch sizes.

[3] https://www.anthropic.com/news/claude-opus-4-7