CHARBENCH: Evaluating the Role of Tokenization in Character-Level Tasks

Omri Uzan[1], **Yuval Pinter**[2]

[1]Stanford University, Stanford, CA, USA
[2]Faculty of Computer and Information Science, Ben-Gurion University of the Negev, Beer Sheva, Israel
uzan@stanford.edu

Abstract

Tasks that require character-level reasoning, such as counting or locating characters within words, remain challenging for contemporary language models. A common conjecture is that language models' reliance on subword units, rather than characters, contributes to their struggles with character-level tasks, yet recent studies offer conflicting conclusions about the role of tokenization, leaving its impact unclear. To address this gap, we introduce CHARBENCH, a comprehensive benchmark of character-level tasks that is two orders of magnitude larger than existing alternatives. We evaluate a diverse range of leading open-weight and proprietary models on CHARBENCH and find that it presents a significant challenge to modern LLMs, with average accuracies of 43.6% and 32.3% on some tasks. We present an in-depth analysis of how intrinsic properties of words and their segmentations into tokens correspond to model performance. For counting tasks, we find that tokenization properties are weakly correlated with correctness, while the length of the queried word and the actual character count play a more significant part. In contrast, for tasks requiring intra-word positional understanding, performance is negatively correlated with the length of the token containing the queried character, suggesting that longer tokens obscure information on character position for LLMs. We encourage future work to build on the benchmark and evaluation methodology introduced here as tools for improving model performance on these tasks.

Code — https://github.com/omriuz/CharBench
Datasets —
https://huggingface.co/datasets/omriuz/CharBench

Introduction

In recent years, the disconnection between large language models (LLMs)' stellar performance on high-level tasks requiring deep understanding of language and their wholly underwhelming ability to perform low-level, even menial tasks at the surface of text analysis, has garnered much attention. One canonical example of such low-level tasks where LLMs appear to fail spectacularly and unexpectedly is the character counting question, such as *How many 'r's are there in* `strawberry?` or *How many 'n's are in* `mayonnaise?`, while such questions are trivially solvable by humans.

While these weaknesses can be mitigated to some extent by prompt engineering or tool usage, for instance by instructing the model to spell or by invoking a code interpreter, investigating why such failures occur can offer deeper insight into fundamental limitations of modern architectures and the inductive biases embedded within them.

One common conjecture for explaining models' weak performance on character tasks is that language models operate on *subword units* rather than individual characters, and that this mismatch plays a key role in their struggles (as illustrated in Figure 1). Prior studies have examined this issue, yet have reached conflicting conclusions about the role of tokenization (Wang et al. 2025; Zhang, Cao, and You 2024; Shin and Kaneko 2024; Xu and Ma 2025; Fu et al. 2024), leaving researchers perplexed about its true impact on this type of problem solving.

In order to rigorously investigate this phenomenon, we present CHARBENCH, a comprehensive suite of evaluation tasks designed to systematically assess the performance of LLMs on character-level reasoning tasks in English. CHARBENCH is designed to test LLMs' ability to reason over characters, with separate tasks for assessing models' performance on positional understanding (indexing) and occurrence tracking (counting). We evaluate a diverse set of state-of-the-art language models from both proprietary and open-weight architectures over a range of parameter sizes. CHARBENCH proves to be a substantial challenge, with an average accuracy of 50.3% across models. Positional understanding appears to be particularly difficult, with average accuracies of 43.6% and 32.4% in the tasks in this category.

To evaluate the role of tokenization at scale, we build on a line of work investigating intrinsic tokenizer metrics (Gallé 2019; Zouhar et al. 2023; Uzan et al. 2024). We observe a consistent linear decline in performance as word length increases, echoing previous findings (Fu et al. 2024) and underscoring a fundamental limitation of current LLMs in processing longer strings. For counting tasks, we find that performance is strongly correlated with the actual number of character occurrences, while tokenization properties show weaker correlations. In contrast, for tasks that require locating specific characters, the length of the token containing the target character is the most correlated feature. We find that accuracy declines as the target token length increases, revealing a tradeoff between token-level compression and the

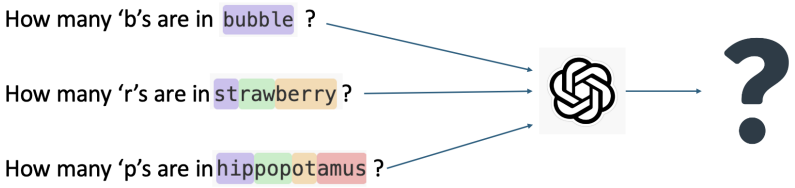


Figure 1: Examples of different subword token segmentations for GPT-4o prompts on character counting tasks. Do the properties of words and their segmentation into tokens correlate with model performance?

model's ability to reason about character-level structure. Notably, while much tokenization work has focused on properties such as the number of tokens per word and average token length (Gallé 2019; Goldman et al. 2024; Schmidt et al. 2024; Beinborn and Pinter 2023), our findings suggest that, for character-level tasks, neither plays a significant role in model performance, and other properties correlate more strongly.

Our findings add a new perspective to the debate in the literature about the role of tokenization in character-level tasks, and complement prior work that frames tokenization as either crucial (Wang et al. 2025; Zhang, Cao, and You 2024; Shin and Kaneko 2024) or insignificant (Xu and Ma 2025; Fu et al. 2024). We show that its relationship with surface form is substantial for certain tasks, and less meaningful for others. We hope this benchmark and evaluation framework serves as a foundation for future work aimed at better understanding and improving model performance on these tasks.

Related Work

Intrinsic Measures of Tokenization. Prior work has investigated intrinsic properties of tokenization, primarily focusing on tokenizer behavior across word sequences. Gallé (2019) argue that the effectiveness of the byte-pair encoding algorithm (BPE; Sennrich, Haddow, and Birch 2016) in translation tasks stems from its ability to represent word sequences with fewer symbols. Zouhar et al. (2023) show that Rényi entropy over token distributions across word sequences correlates more strongly with downstream performance than token count alone. However, subsequent studies provide counterexamples where optimizing for token count or Rényi entropy does not lead to improved downstream performance (Cognetta et al. 2024; Schmidt et al. 2024). Alongside information theory-based measures, Uzan et al. (2024) also integrate intrinsic metrics from cognitive science (Beinborn and Pinter 2023) and morphology (Gow-Smith et al. 2022) into a unified benchmark. Nonetheless, these approaches predominantly assess tokenizer performance at the word-sequence level, overlooking intra-word dynamics that are informative for character-level tasks.

Benchmarking Character-Level Performance in LLMs. While other benchmarks for character-level tasks have been proposed, none that we know of are well-suited for large-scale statistical evaluation of tokenizer properties in relation to downstream performance. Efrat, Honovich, and Levy (2023) introduce LMENTRY, which includes some character-level tasks, but these primarily target the first or last character in words, limiting their ability to generalize to broader tokenization behavior. Chai et al. (2024) present a benchmark with several token-structure probing tasks, containing roughly 300 words, two orders of magnitude below our CHARBENCH. CUTE (Edman, Schmid, and Fraser 2024) provides a broader set of challenging tasks, yet it is intentionally limited to mostly frequent, single-token words. Wang et al. (2025) take a similar approach to ours by automatically generating a large-scale collection of verifiable question-answer pairs involving natural language operations; however, their benchmark primarily targets word-sequence performance rather than character-level or subword-level phenomena.

Tokens and Characters. While the extent to which subword tokens encode character-level information has been thoroughly studied at the embedding level, methods for analyzing this relationship in downstream generation remain limited. Itzhak and Levy (2022) show that subword-based pretrained models can recover correct spellings, suggesting that token embeddings retain some fine-grained character-level information. Kaushal and Mahowald (2022) further support this claim by probing token embeddings for individual characters, attributing this capacity to variation in subword tokenization across related strings during pretraining. More recent work (Xu and Ma 2025; Zhang, Cao, and You 2024) explores the link between subword tokenization and character-level performance but arrives at conflicting conclusions regarding its significance. A key limitation shared by these approaches is their reliance on explicit perturbations, adding special characters to induce character-level segmentation, which introduces noise, diverges from the model's training distribution, inflates input length, and ultimately hinders direct comparability.

We use intrinsic metrics of tokenization that emphasize intra-word dynamics, perform a large-scale statistical evaluation with control for confounding factors, and assess tokenizers' role from a purely statistical perspective without perturbing input strings, allowing us to draw more statistically robust conclusions.

Task	Example	Label
Character Frequency Count	How many times does the character `'s'` appear in the string "`mississippi`"?	4
Unique Character Count	How many unique characters appear in the string `'balloon'`?	5
Find First Occurrence	What is the index of the first occurrence of the character `'e'` in the string `'cheese'`? Start counting from 0.	2
Find Last Occurrence	What is the index of the last occurrence of the character `'n'` in the string `'cinnamon'`? Start counting from 0.	7

Table 1: Example questions from CHARBENCH. One question is shown per task. Words and characters are interchangeable within the prompt template.

CHARBENCH

The core insight in the construction of CHARBENCH is that many character-level reasoning tasks are inherently deterministic, making them well-suited for large-scale, controlled evaluations. While these tasks are presented in natural language, for example, *"How many `r`'s are in 'strawberry'?"*, they can be directly translated into more simple computational operations, such as `count('r', 'strawberry')`. By exploiting this deterministic and verifiable structure, CHARBENCH enables the scalable construction of question-answer pairs while controlling for confounding factors such as word length, frequency, and token count.

To construct CHARBENCH, we developed a set of modular task templates that represent a variety of character-level reasoning scenarios. These templates can be instantiated with arbitrary words and characters, as illustrated in Figure 1. We focused on tasks that are verifiable and can be mapped directly to simple Python functions, such as `count()` or `index()`. To populate the benchmark, we uniformly sampled 175,000 strings from the MiniPile dataset (Kaddour 2023), ensuring balanced coverage across lengths from 4 to 10 characters. This stratified sampling strategy mitigates potential biases related to word length and frequency.

CHARBENCH evaluates two types of character-level information within words: *occurrence information*, which captures the presence of specific characters, and *positional information*, which reflects the locations of characters within a word (e.g., which character appears at a given index). Previous work on character knowledge in token embeddings has primarily examined character occurrence (Kaushal and Mahowald 2022) and relative spelling patterns (Itzhak and Levy 2022), without explicitly evaluating whether models encode information about absolute character positions (while spelling patterns capture relative order, they do not require awareness of absolute positions within a word).

To capture both quantitative and positional aspects of word-level character knowledge, CHARBENCH includes two task categories: *counting* and *indexing*. Counting tasks assess an LLM's ability to determine character frequencies within strings, simple for humans on short words but often challenging for language models. These include counting the occurrences of a specific character and identifying the set of unique characters in a word. Indexing tasks evaluate positional understanding by requiring the model to locate the position (index) of either the first or last occurrence of a specified character within a string. While it is relatively simple to construct more tasks, we focus on two tasks per aspect to enable simpler qualitative analysis of models' performance.

Experiments

We evaluate a diverse set of state-of-the-art language models on CHARBENCH. We include both proprietary and open-weight architectures, selected based on the availability of *open tokenizer access*, which is critical for our analysis of the interaction between a model and the tokenizer it has access to. All selected models use the widely-adopted byte-pair encoding (BPE) tokenization scheme.

Open-weight Models. We evaluate several prominent open-weight models: DeepSeek-V3 (DeepSeek-AI et al. 2025), Llama-3.3-70B, Llama-3.1-405B (Grattafiori et al. 2024), and Mistral-7B (Jiang et al. 2023). These were selected to represent a range of model sizes and architectural designs. All models are accessed via the `Together.AI` API,[1] ensuring a consistent inference environment and enabling evaluation of large-scale models that would otherwise require substantial local compute resources.

Proprietary Models. We include OpenAI's GPT-4o (OpenAI et al. 2024), GPT-4o-mini, and GPT-3.5-turbo in our evaluation, as these models provide unrestricted access to their tokenization mechanism via the `tiktoken` library.[2] Other state-of-the-art proprietary models were excluded due to restrictions on tokenizer access, which prevents an analysis of tokenization effects.

Evaluation Protocol. Model performance is quantified using accuracy, defined as the proportion of predictions matching the gold-standard label (exact match).

[1] https://www.together.ai/

[2] https://github.com/openai/tiktoken

Model	Overall	count_char	count_unique	find_first	find_last
GPT-4o	**70.73%**	**89.45%**	**65.51%**	**64.65%**	**63.14%**
GPT-4o-mini	49.99%	82.54%	47.32%	42.89%	27.07%
GPT-3.5-turbo	45.22%	79.11%	52.08%	28.96%	20.72%
DeepSeek-V3	57.18%	74.52%	57.31%	57.85%	38.72%
Llama-3.3-70B	39.98%	61.90%	43.07%	34.11%	20.74%
Meta-Llama-3.1-405B	55.49%	81.21%	44.83%	54.24%	41.01%
Mistral-7B	34.72%	71.65%	30.63%	20.76%	15.89%
Average	50.33%	77.34%	47.96%	43.64%	32.37%

Table 2: Model results across the different tasks in CHARBENCH. Tasks are identified by their prefixes and are ordered consistently with Table 1.

Parameter Settings. To ensure fair comparison and reproducibility, we standardize all evaluation settings across models. We set the temperature to 0 to eliminate sampling variability and produce deterministic outputs; we use uniform system and user prompt templates for all models; we maintain consistent API usage and parameter settings across both open-weight and proprietary models.

Results

Model Performance. CHARBENCH presents a substantial challenge for modern language models, with an average accuracy of 50.33% across all evaluated models. GPT-4o achieves the highest performance by a significant margin, attaining an average accuracy of 70.73% and outperforming all other models across tasks. Among open-weight models, DeepSeek-V3 achieves the strongest overall performance, outperforming others across most benchmark tasks. Llama-3.1-405B follows closely, and surpasses DeepSeek-V3 on the `count_char` and `find_last` tasks. Notably, Mistral-7B trails Llama-3.3-70B by only 5 points on average, despite being 10 times smaller in parameter count, and even surpasses it by 10 points on the `count_char` task. We also find that GPT-4o-mini exhibits a substantial drop in character-level performance compared to GPT-4o, trailing by an average of 20 points. Complete results are provided in Table 2.

Task Performance. Tasks that require absolute positional understanding over characters, namely `find_first` and `find_last`, prove significantly more challenging for models than counting tasks. While many prior works have focused on character-level understanding through character occurrence, our results indicate that models struggle more with tracking character positions. We consistently observe a performance gap between `find_first` and `find_last`, with the former being solved more reliably. This gap also varies significantly across models, showing an almost 20% difference for DeepSeek-V3, compared to just 0.5% for GPT-4o. Notably, the canonical question *"How many 'r's are there in* `strawberry`*?"* (`count_char`), although still far from solved, emerges as the easiest task for models on CHARBENCH, by a substantial margin.

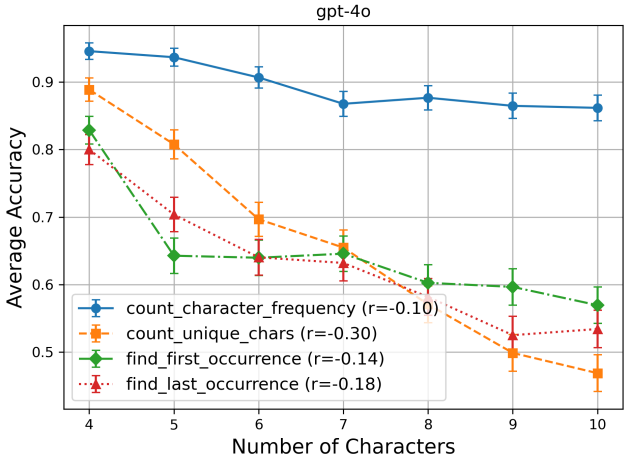


Figure 2: GPT-4o's accuracy on all four tasks as a function of word length, number of tokens, and character spread on the *count character frequency* task.

Analysis

We examine how model performance on CHARBENCH correlates with intrinsic properties of the queried word and its tokenization, as well as task-specific attributes such as the gold truth and the queried character when applicable. We treat each prediction as a binary indicator (1 = correct, 0 = incorrect); we quantify its association with intrinsic properties using Pearson's Correlation. We examine the following properties:

Word Length (WL), the number of characters in the word.

Gold Truth (GT), the correct answer for the given question. In indexing tasks, this is the index of the queried character. In counting tasks, this is the true count of the target character(s).

Gold Truth Divided by Word Length (GT/WL), a normalized variant of GT. In indexing tasks, it reflects the relative position of the queried character within the word (i.e., small values for the start of the word and high values for the ending). In the `Character Frequency Count` task, it indicates the proportion of the word occupied by

task	model	WL	GT	GT/WL	NT	CR	TTL
Find First Occurrence	Llama-405B	–0.212	**-0.325**	–0.277	0.004	–0.191	–0.317
	Llama-70B	–0.257	–0.237	–0.167	–0.010	–0.208	**-0.275**
	DeepSeek-V3	–0.184	**-0.542**	–0.511	–0.025	–0.152	–0.291
	Mistral-7B	–0.134	**-0.259**	–0.247	–0.022	–0.095	–0.048
	GPT-4o-mini	–0.147	–0.483	**-0.497**	–0.006	–0.139	–0.242
	GPT-3.5	**-0.223**	–0.126	–0.041	–0.003	–0.175	–0.194
	GPT-4o (best-performing)	–0.136	–0.170	–0.128	0.059	–0.158	**-0.245**
	Average	–0.185	**-0.306**	–0.267	0.000	–0.160	–0.230
Find Last Occurrence	Llama-405B	–0.247	–0.092	–0.017	–0.028	–0.197	**-0.267**
	Llama-70B	–0.168	0.279	**0.433**	–0.050	–0.109	–0.080
	DeepSeek-V3	**-0.267**	–0.126	–0.042	–0.041	–0.205	–0.242
	Mistral-7B	**-0.099**	0.009	0.086	–0.015	–0.067	–0.019
	GPT-4o-mini	–0.201	0.258	**0.420**	–0.067	–0.127	–0.161
	GPT-3.5	–0.135	0.295	**0.468**	–0.094	–0.048	–0.010
	GPT-4o (best-performing)	–0.180	–0.126	–0.066	0.020	–0.185	**-0.259**
	Average	**-0.185**	0.071	0.183	–0.039	–0.134	–0.148
Character Frequency Count	Llama-405B	–0.242	**-0.478**	–0.244	–0.035	–0.183	–
	Llama-70B	**-0.364**	–0.011	0.212	–0.099	–0.219	–
	DeepSeek-V3	–0.125	**-0.205**	–0.091	–0.025	–0.088	–
	Mistral-7B	–0.240	**-0.531**	–0.319	–0.106	–0.099	–
	GPT-4o-mini	–0.194	**-0.359**	–0.175	–0.015	–0.175	–
	GPT-3.5	–0.201	**-0.228**	–0.090	–0.067	–0.106	–
	GPT-4o (best-performing)	–0.099	**-0.340**	–0.243	0.009	–0.083	–
	Average	–0.209	**-0.307**	–0.136	–0.048	–0.136	–
Unique Character Count	Llama-405B	–0.433	–0.122	**0.669**	–0.189	–0.222	–
	Llama-70B	–0.488	–0.198	**0.642**	–0.193	–0.269	–
	DeepSeek-V3	**-0.317**	–0.271	0.114	–0.106	–0.195	–
	Mistral-7B	–0.262	**-0.364**	–0.136	–0.048	–0.178	–
	GPT-4o-mini	–0.517	–0.213	**0.663**	–0.243	–0.257	–
	GPT-3.5	–0.364	**-0.366**	0.102	–0.116	–0.230	–
	GPT-4o (best-performing)	–0.301	–0.099	**0.462**	–0.108	–0.183	–
	Average	**-0.383**	–0.233	0.359	–0.143	–0.219	–

Table 3: Correlation coefficients between correctness and input properties. Abbreviations: WL = word length; GT = gold truth; GT/WL = gold truth divided by word length; NT = number of tokens; CR = compression rate; TTL = target token length. Dashes (–) denote not applicable values (e.g., TTL for counting tasks). Bold indicates the strongest absolute correlation per row.

the target character. In the `Unique Character Count` task, it captures the proportion of unique characters in the word. From each word's segmentation into tokens, we derive the following tokenization-level properties:

Number of Tokens (NT), the total number of tokens the word is split into via tokenization.

Compression Ratio (CR), the ratio of character length to token count, indicating the average number of characters per token.

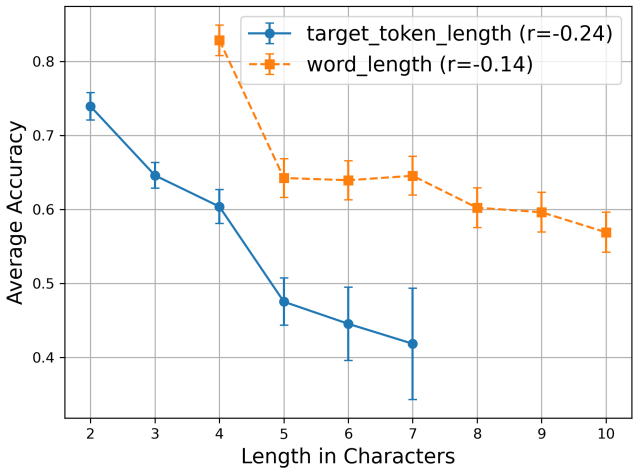
Target Token Length (TTL), The character length of the token that contains the queried character. This is defined only for indexing tasks that reference a specific character position.

Table 3 reports, for each CHARBENCH task and model, the correlation between model performance and each intrinsic property. For aggregation, it presents two complementary indicators: the *average correlation* across all models, which reflects general trends, and the *correlation for the best-performing model* on the benchmark, GPT-4o in all tasks, which highlights how properties relate to performance in a strong model. To better illustrate the relationship between performance and these intrinsic properties, we analyze the average performance across property values for the best-performing model on CHARBENCH, GPT-4o. In figures below, we plot the mean accuracy for each property value, with error bars showing standard error computed from the bucket.

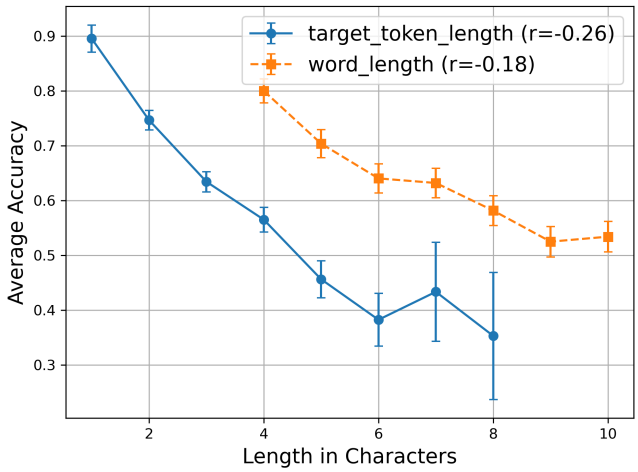
Word Length Consistently Correlates Negatively with Performance

Word length exhibits a consistent correlation with performance across models and tasks, echoing findings by Fu et al. (2024).

Figure 2 illustrates this relationship for GPT-4o, showing



(a) Find First Occurrence



(b) Find Last Occurrence

Figure 3: Accuracy as a function of word length, plotted separately for each task. Points show the mean accuracy for all items of a given length. Error bars indicate tandard error of the binomial proportion for that bucket.

that longer words are consistently associated with lower average prediction accuracy. As shown in Table 3, the `Unique Character Count` task displays the strongest negative correlation with word length, with an average correlation of -0.383 across models.

Other tasks exhibit weaker correlations, generally ranging between -0.18 and -0.21. This supports the notion that tasks requiring holistic processing of the word, such as `Unique Character Count`, are more sensitive to word length than tasks focused on localized character-level operations.

The Gold Truth Effects

Figure 4 compares the relationship between GPT-4o's performance on the `Character Frequency Count` task and three intrinsic properties: word length, number of tokens, and the gold truth. We find that performance on this task for GPT-4o is uncorrelated with the total number of tokens and only weakly negatively correlated with word length (-0.10). In contrast, it shows a notably stronger negative correlation with the **Gold Truth (GT)** metric, which measures how many such characters are present in the word. This trend is consistent across all models except Llama-70B, as reflected in Table 3.

For the `Unique Character Count` task, we find that the gold truth normalized by the length of the word **(GT/WL)** is positively correlated with performance, with correlation values of over 0.64 for several models. This suggests that as more characters in the word are unique, so it is easier for the model to predict the answer.

For indexing tasks, the **GT** and **GT/WL** are also correlated with the average performance, yet mostly so for the weaker models. The best performing model (GPT-4o) is less correlated with the gold truth variants.

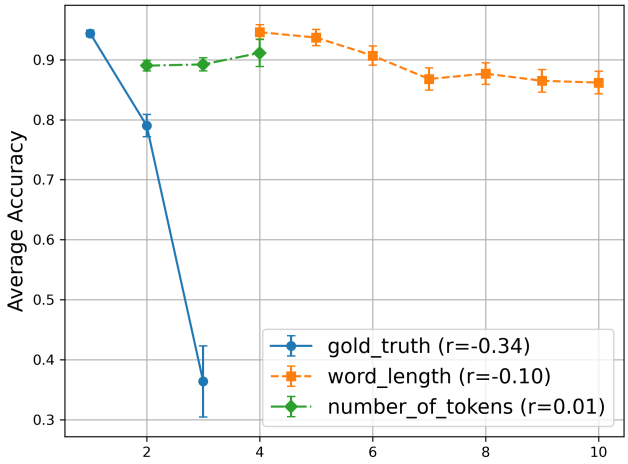
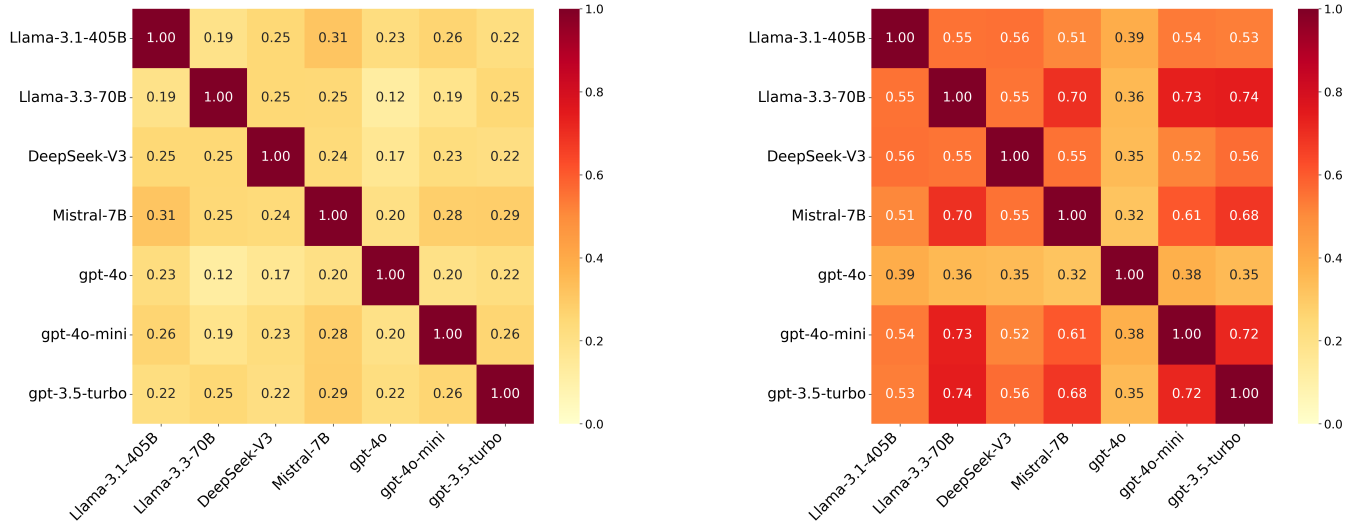


Figure 4: Accuracy as a function of word length, number of tokens, and character spread on the *count character frequency* task.

Longer Tokens Obscure Character Position Information for LLMs

We analyze the effect of token length on model performance on indexing tasks, which require identifying a specific character in the input. These tasks inherently depend on correctly locating the token that contains the target character. For the best-performing model, GPT-4o, we find that the length of this *target token* is the most strongly correlated feature with accuracy.

Figure 3 presents the average performance as a function of both word length and target token length. We can see that performance declines sharply as the target token becomes longer. This finding is notable because tokenizers are typically optimized to compress character sequences into



(a) Count Character Frequency

(b) Find Last Occurrence

Figure 5: Error overlap between models, measured as the intersection-over-union of incorrectly answered questions for each model pair.

fewer tokens to improve computational efficiency. Our analysis suggest a tradeoff between token-level compression and the model's ability to reason about character-level positions within words.

Number of Tokens and Compression Ratio Are Weakly Correlated with Performance

Across tasks and models, Table 3 shows that both the number of tokens a word is split into and the compression ratio (i.e., the average token length) exhibit weaker correlations with model performance compared to other intrinsic properties. As compression is a property receiving much focus in work about tokenization, our findings contribute to the discussion by suggesting that, for character-level tasks, neither the number of tokens nor the degree of compression within a token plays a dominant role in model accuracy.

Error Analysis

Error Overlap between Different Models. We compute the intersection-over-union of incorrectly answered questions for each task and pair of models. Figure 5 presents these ratios for the `Character Frequency Count` and `Find Last Occurrence` tasks. For the former, overlap between models is negligible, suggesting that errors do not follow a shared pattern. In contrast, the latter exhibits substantial overlap across most models, indicating more consistent failure cases. Notably, GPT-4o is an outlier and shows minimal overlap with the other models.

Counting Bias. We also measure if there exists a systematic bias in the models' predictions. For example, do the models tend to overcount in counting tasks, or over-index in indexing tasks?

We find that there is large variance across models and tasks. In general, on the *find_last_occurrence* task, models tend to over-index (which can be useful as a guessing bias), whereas on the *count_character_frequency* task, models tend to undercount.

Mixed-Case Effects. In the evaluation prompt, models are explicitly instructed to treat upper-case and lower-case as different characters for the purpose of the question. We examine whether this aspect of upper/lower case has any effect on model performance. We find that the presence of mixed case characters appears to introduce a measurable bias, yet no a catastrophic one. In fact, for some models and tasks, mixed-case inputs seem to improve performance.

Conclusion

In this work, we introduced a large-scale benchmark for evaluating subword phenomena in character-level tasks. Evaluating tokenization effects in models remains both computationally and architecturally challenging, requiring innovative approaches to conduct such analysis reliably at scale. We hope the benchmark and evaluation measures presented here will support broader efforts to improve model performance on these seemingly simple tasks.

Limitations

Our evaluation is limited to character-level tasks in English, chosen to minimize potential confounders such as frequency in the training data. We encourage future work to replicate these experiments in additional languages, prioritizing diversity in typology and script.

Acknowledgments

We would like to thank anonymous reviewers for many helpful suggestions. This research was supported by the Israel Science Foundation (grant No. 1166/23).

References

Beinborn, L.; and Pinter, Y. 2023. Analyzing Cognitive Plausibility of Subword Tokenization. In Bouamor, H.; Pino, J.; and Bali, K., eds., *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, 4478–4486. Singapore: Association for Computational Linguistics.

Chai, Y.; Fang, Y.; Peng, Q.; and Li, X. 2024. Tokenization Falling Short: On Subword Robustness in Large Language Models. In Al-Onaizan, Y.; Bansal, M.; and Chen, Y.-N., eds., *Findings of the Association for Computational Linguistics: EMNLP 2024*, 1582–1599. Miami, Florida, USA: Association for Computational Linguistics.

Cognetta, M.; Zouhar, V.; Moon, S.; and Okazaki, N. 2024. Two Counterexamples to Tokenization and the Noiseless Channel. In Calzolari, N.; Kan, M.-Y.; Hoste, V.; Lenci, A.; Sakti, S.; and Xue, N., eds., *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, 16897–16906. Torino, Italia: ELRA and ICCL.

DeepSeek-AI; Liu, A.; Feng, B.; Xue, B.; Wang, B.; Wu, B.; Lu, C.; Zhao, C.; Deng, C.; Zhang, C.; and Others. 2025. DeepSeek-V3 Technical Report. arXiv:2412.19437.

Edman, L.; Schmid, H.; and Fraser, A. 2024. CUTE: Measuring LLMs' Understanding of Their Tokens. In Al-Onaizan, Y.; Bansal, M.; and Chen, Y.-N., eds., *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, 3017–3026. Miami, Florida, USA: Association for Computational Linguistics.

Efrat, A.; Honovich, O.; and Levy, O. 2023. LMentry: A Language Model Benchmark of Elementary Language Tasks. In Rogers, A.; Boyd-Graber, J.; and Okazaki, N., eds., *Findings of the Association for Computational Linguistics: ACL 2023*, 10476–10501. Toronto, Canada: Association for Computational Linguistics.

Fu, T.; Ferrando, R.; Conde, J.; Arriaga, C.; and Reviriego, P. 2024. Why Do Large Language Models (LLMs) Struggle to Count Letters? arXiv:2412.18626.

Gallé, M. 2019. Investigating the Effectiveness of BPE: The Power of Shorter Sequences. In Inui, K.; Jiang, J.; Ng, V.; and Wan, X., eds., *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, 1375–1381. Hong Kong, China: Association for Computational Linguistics.

Goldman, O.; Caciularu, A.; Eyal, M.; Cao, K.; Szpektor, I.; and Tsarfaty, R. 2024. Unpacking Tokenization: Evaluating Text Compression and its Correlation with Model Performance. In Ku, L.-W.; Martins, A.; and Srikumar, V., eds., *Findings of the Association for Computational Linguistics: ACL 2024*, 2274–2286. Bangkok, Thailand: Association for Computational Linguistics.

Gow-Smith, E.; Tayyar Madabushi, H.; Scarton, C.; and Villavicencio, A. 2022. Improving Tokenisation by Alternative Treatment of Spaces. In Goldberg, Y.; Kozareva, Z.; and Zhang, Y., eds., *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, 11430–11443. Abu Dhabi, United Arab Emirates: Association for Computational Linguistics.

Grattafiori, A.; Dubey, A.; Jauhri, A.; Pandey, A.; Kadian, A.; Al-Dahle, A.; Letman, A.; Mathur, A.; Schelten, A.; Vaughan, A.; and Others. 2024. The Llama 3 Herd of Models. arXiv:2407.21783.

Itzhak, I.; and Levy, O. 2022. Models In a Spelling Bee: Language Models Implicitly Learn the Character Composition of Tokens. In Carpuat, M.; de Marneffe, M.-C.; and Meza Ruiz, I. V., eds., *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 5061–5068. Seattle, United States: Association for Computational Linguistics.

Jiang, A. Q.; Sablayrolles, A.; Mensch, A.; Bamford, C.; Chaplot, D. S.; de las Casas, D.; Bressand, F.; Lengyel, G.; Lample, G.; Saulnier, L.; Lavaud, L. R.; Lachaux, M.-A.; Stock, P.; Scao, T. L.; Lavril, T.; Wang, T.; Lacroix, T.; and Sayed, W. E. 2023. Mistral 7B. arXiv:2310.06825.

Kaddour, J. 2023. The MiniPile Challenge for Data-Efficient Language Models. arXiv:2304.08442.

Kaushal, A.; and Mahowald, K. 2022. What do tokens know about their characters and how do they know it? In Carpuat, M.; de Marneffe, M.-C.; and Meza Ruiz, I. V., eds., *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2487–2507. Seattle, United States: Association for Computational Linguistics.

OpenAI; Hurst, A.; Lerer, A.; Goucher, A. P.; Perelman, A.; Ramesh, A.; Clark, A.; Ostrow, A.; Welihinda, A.; Hayes, A.; and Others. 2024. GPT-4o System Card. arXiv:2410.21276.

Schmidt, C. W.; Reddy, V.; Zhang, H.; Alameddine, A.; Uzan, O.; Pinter, Y.; and Tanner, C. 2024. Tokenization Is More Than Compression. In Al-Onaizan, Y.; Bansal, M.; and Chen, Y.-N., eds., *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, 678–702. Miami, Florida, USA: Association for Computational Linguistics.

Sennrich, R.; Haddow, B.; and Birch, A. 2016. Neural Machine Translation of Rare Words with Subword Units. In Erk, K.; and Smith, N. A., eds., *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 1715–1725. Berlin, Germany: Association for Computational Linguistics.

Shin, A.; and Kaneko, K. 2024. Large language models lack understanding of character composition of words. *arXiv preprint arXiv:2405.11357*.

Uzan, O.; Schmidt, C. W.; Tanner, C.; and Pinter, Y. 2024. Greed is All You Need: An Evaluation of Tokenizer Inference Methods. In Ku, L.-W.; Martins, A.; and Srikumar, V., eds., *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, 813–822. Bangkok, Thailand: Association for Computational Linguistics.

Wang, X.; Fu, H.; Wang, J.; and Gong, N. Z. 2025. StringLLM: Understanding the String Processing Capability

of Large Language Models. In *The Thirteenth International Conference on Learning Representations*.

Xu, N.; and Ma, X. 2025. LLM The Genius Paradox: A Linguistic and Math Expert's Struggle with Simple Word-based Counting Problems. arXiv:2410.14166.

Zhang, X.; Cao, J.; and You, C. 2024. Counting Ability of Large Language Models and Impact of Tokenization. arXiv:2410.19730.

Zouhar, V.; Meister, C.; Gastaldi, J.; Du, L.; Sachan, M.; and Cotterell, R. 2023. Tokenization and the Noiseless Channel. In Rogers, A.; Boyd-Graber, J.; and Okazaki, N., eds., *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 5184–5207. Toronto, Canada: Association for Computational Linguistics.