Token Efficiency of OpenAI GPT-5 Models for English and Turkish

Mutlu Doğuş Yıldırım

dogus@ileri.org.tr

December 6, 2025

Abstract

We present an initial quantitative analysis of token efficiency for OpenAI GPT-5.1 and GPT-5-mini on English and Turkish text. Using a controlled "echo" task on 100 parallel English–Turkish sentence pairs, we model the number of prompt tokens as a linear function of character length and perform paired comparisons between languages. We find a clear and statistically robust "Turkish overhead": for the same underlying content, Turkish inputs require on average approximately six more input tokens per sentence and about 0.15 more tokens per character than their English counterparts. Removing Turkish diacritics does not reduce token usage; in fact it slightly increases tokens per character. These effects translate directly into higher effective cost for Turkish workloads, with Turkish sentences being roughly 21–22% more expensive than English sentences on the evaluated OpenAI models. The analysis here establishes a clean baseline that can be extended to other vendors and tasks.

1. Introduction

Large language models (LLMs) are priced and constrained in terms of tokens, yet end users reason in terms of characters, words, and sentences. For languages other than English, especially morphologically rich languages such as Turkish, the mapping from characters to tokens is non-trivial and may introduce systematic differences in effective cost. This note reports an empirical study of token efficiency for OpenAI GPT-5.1 and GPT-5-mini on English and Turkish, focusing on three questions: (i) how well a simple linear model explains prompt token usage as a function of input length, (ii) whether Turkish incurs a consistent token overhead relative to English for the same content, and (iii) whether removing Turkish diacritics changes token efficiency.

2. Methods

We use a dataset of 100 sentence IDs, each associated with three variants: English (en), Turkish (tr), and Turkish without diacritics (tr_nodia). For each sentence and variant, we send a single-turn prompt to the OpenAI Responses API with a fixed system instruction and a simple "echo" task: the model is instructed to return exactly the user-provided text. We evaluate two models: GPT-5.1 and GPT-5-mini. Both use the same temperature and maximum output token settings, but differ in reasoning configuration: GPT-5.1 is run with reasoning disabled, while GPT-5-mini is run with minimal reasoning effort, reflecting the constraints of the respective APIs.

For every request, we record the number of input tokens (prompt_tokens), the number of output tokens (completion_tokens), the total tokens, the response time, and the effective cost computed from the advertised price per million tokens.

Because each sentence ID appears in all three variants for both models, the design naturally supports paired comparisons (e.g., Turkish vs English for the same content). We focus on input-side behavior, modeling prompt_tokens as a linear function of the character length (chars) of the user message. We then compute paired differences for tokens per character and total prompt tokens between variants, together with approximate 95% confidence intervals.

3. Results

3.1 Linear model of prompt tokens vs character length

For each model and variant, we fit a linear regression of the form prompt_tokens ≈ a + b · chars. The table below summarizes the fitted intercept a, slope b (tokens per character), and coefficient of determination R^2.

Model	Variant	n	Mean chars	Mean prompt tokens	Intercept a	Slope b (tokens/char)	R^2
gpt-5-mini	en	100	65.1	43.3	32.0	0.173	0.885
gpt-5-mini	tr	100	64.1	49.2	32.2	0.265	0.945
gpt-5-mini	tr_nodia	100	64.1	50.1	31.9	0.283	0.954

gpt-5.1	en	100	65.1	43.3	32.0	0.173	0.885
gpt-5.1	tr	100	64.1	49.2	32.2	0.265	0.945
gpt-5.1	tr_nodia	100	64.1	50.1	31.9	0.283	0.954

Across all conditions, the intercept is approximately 32 tokens, reflecting a fixed per-call overhead. The slope is approximately 0.173 tokens/character for English, 0.265 tokens/character for Turkish, and 0.283 tokens/character for Turkish without diacritics. R^2 values above 0.88 indicate that this simple linear model captures most of the variance in prompt token counts.

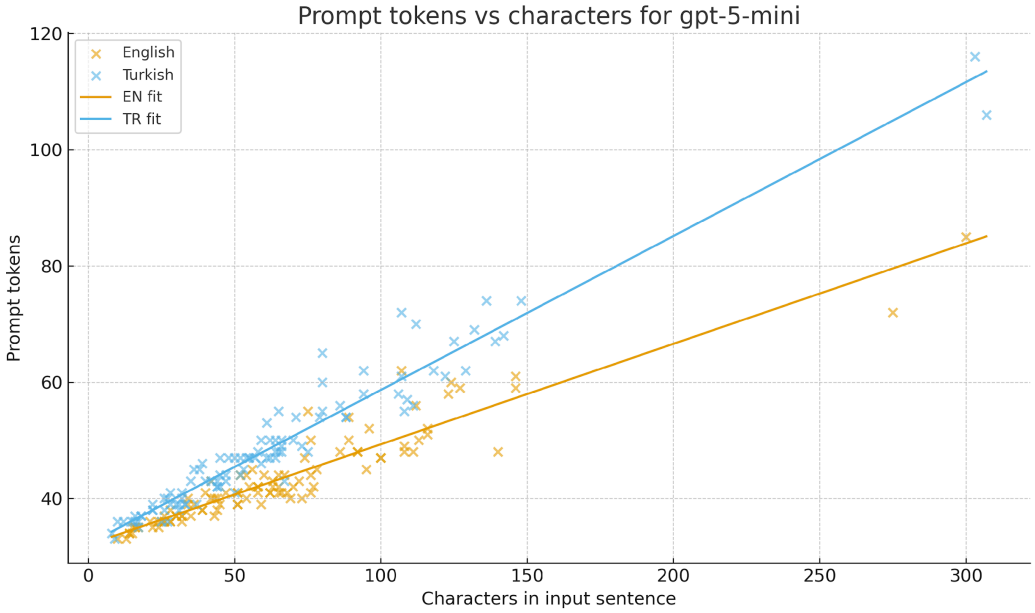

Figure 1. Prompt tokens as a function of character length for GPT-5.1, for English (EN) and Turkish (TR). Points show individual sentences; lines show linear fits.

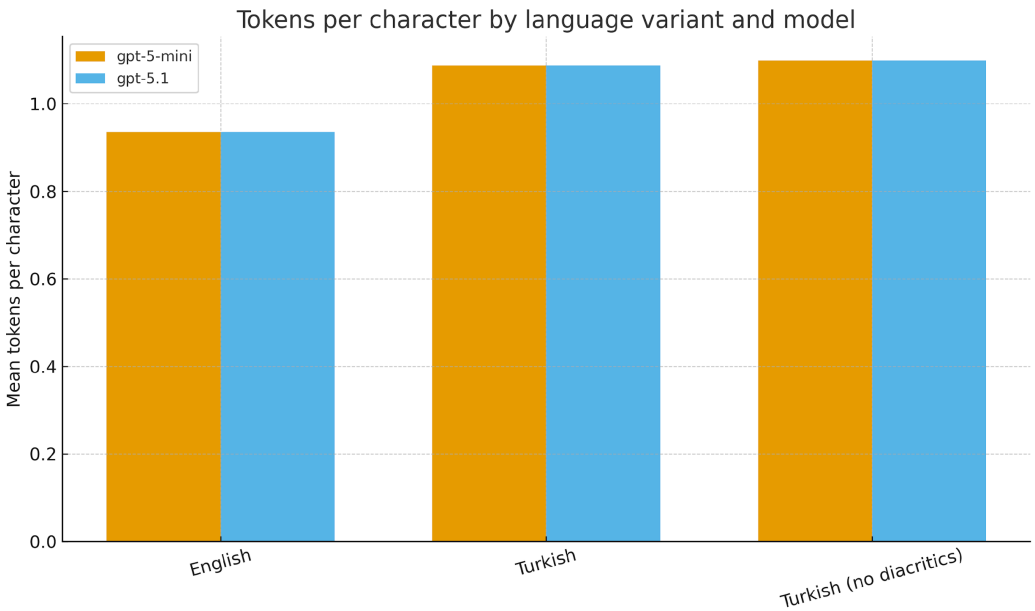

Figure 2. Mean tokens per character for English, Turkish, and Turkish without diacritics (TR-NODIA) across GPT-5.1 and GPT-5-mini.

3.2 Paired comparison: English vs Turkish

Because each sentence appears in both English and Turkish, we compute paired differences per ID. For each model, we consider Δ(tokens/char) = (tokens/char)_TR − (tokens/char)_EN and Δ(prompt_tokens) = prompt_tokens_TR − prompt_tokens_EN.

Model	n pairs	Mean Δ tokens/ char	SD Δ tpc	95% CI Δ tpc	Mean Δ prompt_ tokens	SD Δ pt	95% CI Δ pt
gpt-5-mini	100	0.152	0.264	[0.100, 0.204]	5.880	5.704	[4.748, 7.012]
gpt-5.1	100	0.152	0.264	[0.100, 0.204]	5.880	5.704	[4.748, 7.012]

For both GPT-5.1 and GPT-5-mini, the mean Δ(tokens/char) is approximately 0.152, with a 95% confidence interval around [0.100, 0.204]. In absolute terms, Turkish requires on average about 5.9 additional input tokens per sentence compared to English, with a 95% confidence interval of roughly [4.8, 7.0]. Given average input lengths around 64–65 characters, this translates to a 14–16% increase in prompt tokens for Turkish for the same content.

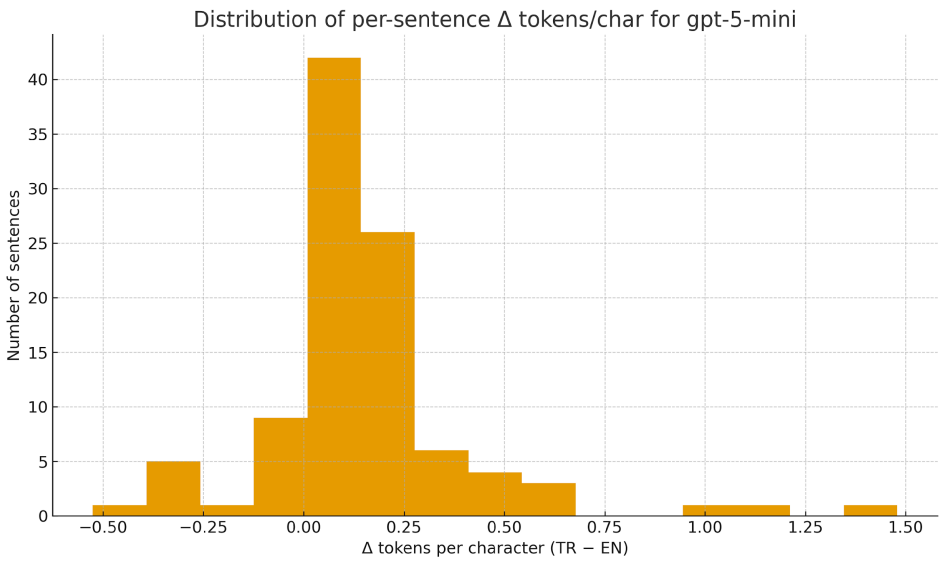

Figure 3. Distribution of per-sentence differences in tokens per character (Δ = TR − EN) for gpt-5-mini. Most sentences show a positive Turkish overhead.

3.3 Effect of removing diacritics

We also compare Turkish with and without diacritics using paired differences per sentence: Δ(tokens/char) = (tokens/char)_TR_NODIA − (tokens/char)_TR.

Model	n pairs	Mean Δ tokens/char	SD Δ tpc	95% CI Δ tpc
gpt-5-mini	100	0.011	0.027	[0.006, 0.016]
gpt-5.1	100	0.011	0.027	[0.006, 0.016]

For both models, the mean Δ(tokens/char) is approximately +0.011, with 95% confidence intervals around [+0.006, +0.017]. Thus, removing Turkish diacritics slightly increases tokens per character rather than decreasing it. From a token efficiency perspective, there is no benefit to degrading Turkish orthography for these OpenAI models.

3.4 Cost implications

Because cost is linear in token counts under the OpenAI pricing scheme, the observed token overhead for Turkish translates directly into higher effective cost.

Model	Mean cost EN	Mean cost TR	TR / EN cost ratio

gpt-5-mini	0.000049	0.000059	1.217
gpt-5.1	0.000285	0.000344	1.206

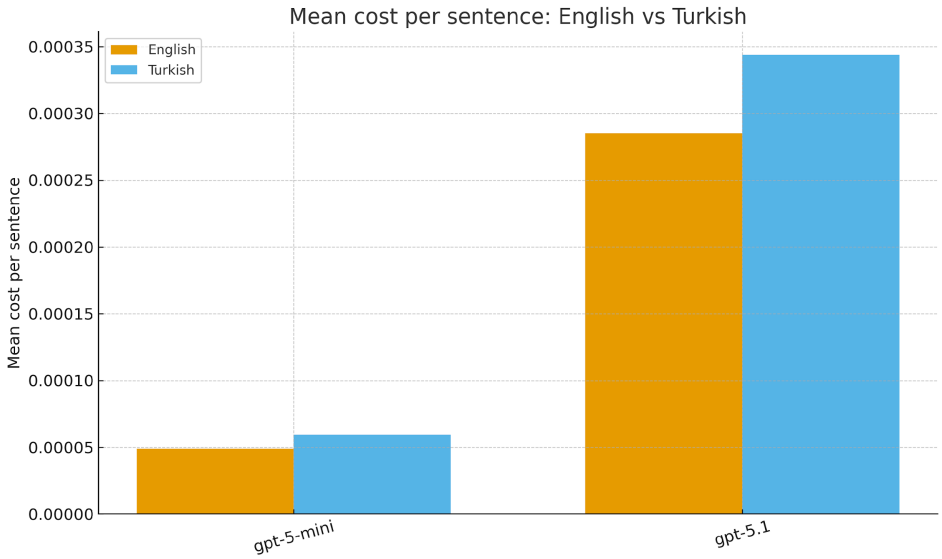

Figure 4. Mean cost per sentence for English (EN) and Turkish (TR) for GPT-5.1 and GPT-5-mini, using the pricing scheme at the time of the experiment.

In this experiment, Turkish sentences are approximately 1.21–1.22 times as expensive as English sentences for both GPT-5.1 and GPT-5-mini. This cost multiplier is a direct consequence of the increased token counts for Turkish inputs and slightly longer outputs, not of any difference in list pricing between languages.

4. Discussion

The analysis reveals a clear and consistent Turkish overhead in token usage on the evaluated OpenAI models. At a mechanistic level, the linear model shows that each additional character in English contributes roughly 0.17 prompt tokens, whereas each additional character in Turkish contributes roughly 0.26–0.28 prompt tokens. This effect is strong enough to be visible with only 100 sentence pairs and is statistically well separated from zero in paired tests.

Practically, this means that developers building Turkish-heavy applications on OpenAI GPT-5.x pay a non-trivial effective premium in token usage and cost compared to equivalent English workloads. While a 20% difference per sentence may appear modest, it compounds at scale: for millions of requests per day, the choice of language materially affects infrastructure cost. Importantly, attempts to reduce tokens by stripping Turkish

diacritics not only harm readability but also fail to deliver token savings; if anything, they increase tokens per character slightly.

These findings highlight the importance of language-aware benchmarking when planning LLM deployments. Headline prices per million tokens do not fully capture the effective cost of serving non-English users; tokenization behavior plays a central role. The methodology used here—parallel sentence pairs, simple echo tasks, and per-character linear modeling—can be directly reused to compare other vendors (e.g., Gemini, Claude, Grok) and to investigate whether some models are intrinsically more token-efficient for morphologically rich languages.

5. Limitations and Future Work

This study has several limitations. First, it considers only a single task (echoing the input) and a single system instruction; more complex tasks involving summarization, reasoning, or tool use may exhibit different tokenization dynamics. Second, the analysis is restricted to two OpenAI models and a snapshot in time; tokenization algorithms and pricing may evolve. Third, the dataset consists of 100 sentences, which is sufficient to reveal clear effects but does not cover the full diversity of real-world input distributions.

Future work should extend this analysis to multiple providers and tasks, incorporate larger and more diverse datasets, and examine latency behavior under varying concurrency and load. It would also be valuable to jointly study efficiency and quality: in some cases, a model that uses more tokens may deliver sufficiently higher quality to justify the additional cost. Nonetheless, even in isolation, the efficiency results reported here provide a useful quantitative baseline for practitioners working with Turkish and other non-English languages.

6. Conclusion

Using a controlled, parallel English–Turkish dataset and a simple yet expressive linear modeling framework, we have shown that Turkish inputs incur a consistent and measurable token overhead on OpenAI GPT-5.1 and GPT-5-mini. For the same semantic content, Turkish sentences require more tokens, cost more, and offer no easy token savings via orthographic simplifications. These findings underscore the need for language-specific efficiency measurements when selecting and configuring LLMs for production use, and they set the stage for broader cross-vendor comparisons in future work.