Do Diacritics Matter? Evaluating the Impact of Arabic Diacritics on Tokenization and LLM Benchmarks

Go Inoue,[1,2] Bashar Alhafni,[1] Nizar Habash,[1,2] Timothy Baldwin[1]
[1]Mohamed bin Zayed University of Artificial Intelligence
[2]New York University Abu Dhabi
{go.inoue,bashar.alhafni,timothy.baldwin}@mbzuai.ac.ae
nizar.habash@nyu.edu

Abstract

Diacritics are orthographic marks added to letters to specify pronunciation, disambiguate lexical meanings, or indicate grammatical distinctions. Diacritics can significantly influence language processing tasks, especially in languages like Arabic, where diacritic usage varies widely across domains and contexts. While diacritics provide valuable linguistic information, their presence can increase subword fragmentation during tokenization, potentially degrading the performance of NLP models. In this paper, we systematically analyze the impact of diacritics on tokenization and benchmark task performance across major Large Language Models (LLMs). Our results demonstrate that while modern LLMs show robustness to the limited diacritics naturally found in texts, full diacritization leads to substantially increased token fragmentation and degraded performance, highlighting the need for careful handling of diacritics in the future development of Arabic LLMs.[1]

1 Introduction

Diacritics are orthographic marks added to letters to specify pronunciation, disambiguate lexical items, or indicate grammatical distinctions. In morphologically rich languages such as Arabic, diacritics play a critical role in conveying lexical and morphosyntactic information as well. While Arabic diacritics can help resolve ambiguity and enhance comprehension, their prevalence varies across domains: highly prevalent in children's books, religious texts, and poetry, but largely absent in news articles and user-generated content.

From an Arabic NLP perspective, diacritics are double-edged. On one hand, they offer valuable linguistic cues for downstream tasks (Habash et al., 2016; Elgamal et al., 2024; Chen et al., 2024); on the other, their presence introduces orthographic variation that increases subword fragmentation (Gorman and Pinter, 2025). This can lead to inefficiencies in representation and degraded downstream performance. Many Arabic NLP pipelines sidestep this complexity by removing diacritics during preprocessing (Antoun et al., 2020; Inoue et al., 2021) and it remains unclear how robust LLMs are to different degrees of diacritization.

In this work, we present the first systematic study of how Arabic diacritics affect tokenization and performance in LLM benchmarks. We evaluate nine LLMs, including Arabic-centric, multilingual, and proprietary models, on three widely adopted Arabic benchmarks—ArabicMMLU (Koto et al., 2024), ArabCulture (Sadallah et al., 2025), and AraTrust (Alghamdi et al., 2025)—with varying degrees of diacritization: (i) all diacritics removed, (ii) naturally occurring diacritics, and (iii) full diacritization.

Our findings reveal that modern LLMs are generally robust to the sparse diacritics found in natural text; in fact, retaining these *wild* diacritics often yields better performance than removing them. However, fully diacritized input consistently leads to more fragmented tokenization, lower representational similarity, and significantly degraded performance. These results highlight the importance of careful diacritic handling in the future development of Arabic LLMs.

Our contributions are as follows:

- We provide a systematic analysis quantifying how Arabic diacritics affect tokenization patterns in Arabic LLMs.
- We present empirical evidence demonstrating the impact of diacritics on Arabic LLM performance across major benchmarks.
- To our knowledge, this is the first systematic analysis of how different degrees of diacritization affect tokenization and LLM performance in Arabic.

[1]Code and data are available at https://github.com/mbzuai-nlp/do_diacritics_matter.

Findings of the Association for Computational Linguistics: EACL 2026, pages 426–442
March 24-29, 2026

2 Background

Arabic diacritics. Arabic script follows an abjad orthography and consists of two classes of symbols: letters and diacritics. Letters are always written, whereas diacritics are optional. Written Arabic may appear in fully diacritized, partially diacritized, or entirely undiacritized forms. Diacritics are non-spacing marks added to letters to supplement the script with phonological cues. While the Arabic script includes many diacritics (52 in the Unicode standard),[2] the core set used in most Modern Standard Arabic (MSA) contexts consists of nine symbols grouped into four types: vowel, nunation, Shadda, and Dagger Alif. Vowel diacritics represent the three short vowels in Arabic: Fatha ◌َ *a*[3] /a/, Damma ◌ُ *u* /u/, and Kasra ◌ِ *i* /i/, along with the Sukun ◌ْ (silence), which marks the absence of a vowel. Nuntation diacritics: ◌ً *ã*, ◌ٌ *ũ*, and ◌ٍ *ĩ*, indicate indefiniteness and occur only at the end of nominals (nouns, adjectives, and adverbs), representing a short vowel followed by /n/. The Shadda ◌ّ ∼ is a gemination marker that indicates consonant doubling and applies to the consonant it follows. It may co-occur with either a vowel or a nunation diacritic. Finally, the special elongation diacritic ◌ٰ á (aka Dagger Alif) indicates a long /a/ vowel (/ā/).

It is worth noting that the Arabic consonant Hamza (glottal stop /'/) can appear as a letter form ء ' or as a diacritic that attaches to specific letters, such as: أ Â, إ Ǎ, ؤ ŵ, ئ ŷ. Unlike other diacritics, the omission of Hamzas is treated as a spelling error.

Role of Arabic diacritics. In addition to marking phonological cues, Arabic diacritics play a key role in disambiguation at the lexical, morphological, and syntactic levels. Functionally, they are often categorized into lexemic diacritics and inflectional diacritics (Habash and Rambow, 2007; Habash, 2010). Lexemic diacritics distinguish between two or more lexemes. For example, the diacritics in the words كَاتِب *kaAtib* 'writer' and كَاتَب *kaAtab* 'he corresponded', distinguish between the meaning of the words rather than their morphosnytactic variants. Inflectional diacritics, distinguish different inflected forms of the same lexeme. For instance, these three forms of the word كاتب *kaAtb* 'writer' vary in terms of their inflectional case and state: كَاتِبُ *kaAtibu* '[nominative definite]', كَاتِبٌ *kaAtibũ* '[nominative indefinite]', كَاتِبٍ *kaAtibĩ* '[genitive indefinite]'.

Domain	%lines (w/ diac)	%words (w/ diac)	#diacs (per word)
Children	81.4	82.6	3.2
Poetry	81.2	53.8	2.1
Novels	50.8	5.6	1.4
UN	15.6	1.4	1.2
News	13.9	1.3	1.1
ChatGPT	58.1	5.3	1.9

Table 1: Statistics of diacritic usage in the Partially Diacritized Dataset (PDD) of Elgamal et al. (2024).

Diacritic usage. Arabic text appears in varying degrees of diacritization depending on the context, genre, and audience. For example, children's literature, poetry, and religious texts often include extensive diacritic marking, aiding readers with pronunciation and comprehension. On the other hand, news articles and formal texts generally have limited or no diacritics, as proficient readers typically rely on contextual clues without requiring diacritics to disambiguate potential interpretations.

Table 1 presents the domain-wise statistics of diacritic usage from the Partially Diacritized Dataset (PDD) (Elgamal et al., 2024), reporting the percentage of lines containing at least one diacritic (**%lines**), the percentage of Arabic words that include at least one diacritic (**%words**), and the average number of diacritics per diacritized Arabic word (**#diacs**). The Children and Poetry subsets exhibit the highest proportion of lines with diacritics (around 81% of lines), whereas News and UN texts contain substantially fewer lines with diacritics (around 14-15% of lines). The percentage of diacritized words differ significantly across domains, where the Children domain has the highest percentage with 82.6%, while UN and News domains only include 1.3-1.4% diacritized words on average.

In what follows, we use three distinct levels of diacritization, following the terminology of Elgamal et al. (2024): (i) text with no diacritics, or undiacritized text (`undiac`); (ii) text with naturally occurring diacritics, or wild diacritics (`wild`); and (iii) fully diacritized text (`full`).

[2] `https://unicode.org/charts/PDF/U0600.pdf`

[3] Arabic HSB transliteration (Habash et al., 2007).

3 Related Work

Diacritization. Extensive research has been devoted to Arabic diacritization, with approaches evolving from rule-based systems (Debili and Achour, 1998; El-Imam, 2004; Nelken and Shieber, 2005) to feature-based machine learning models (Zitouni et al., 2006; Habash and Rambow, 2007; Roth et al., 2008; Pasha et al., 2014; Darwish et al., 2017), and neural methods (Abandah et al., 2015; Belinkov and Glass, 2015; Rashwan et al., 2015; Mubarak et al., 2019; Zalmout and Habash, 2020; Darwish et al., 2021; Elmallah et al., 2024; Mohamed and Mubarak, 2025).

Beyond diacritization itself, several studies have examined its impact on a range of downstream Arabic NLP tasks, including machine translation (Diab et al., 2007; Alqahtani et al., 2016; Fadel et al., 2019), homograph disambiguation (Alqahtani et al., 2019), language proficiency assessment (Hamed and Zesch, 2018), improving text readability (Esmail et al., 2022; ElNokrashy and AlKhamissi, 2024), text-to-speech synthesis (Ungurean et al., 2008), and automatic speech recognition (Aldarmaki and Ghannam, 2023).

Among this work, prior research has studied how different degrees of diacritization (e.g., full or partial) affect downstream task performance. Diab et al. (2007) and Alqahtani et al. (2016) investigated this effect in the context of Arabic-to-English machine translation. Habash et al. (2016) demonstrated that the degree of diacritization in input text correlates positively with the quality of morphological annotation. Several studies (AlKhamissi et al., 2020; Bahar et al., 2023; Elgamal et al., 2024) showed that providing partial diacritization as input can improve the overall diacritization performance. To the best of our knowledge, no previous work has explored the impact of varying degrees of diacritization on the performance of LLMs.

Tokenization. Several studies have shown that tokenizer selection and configuration play a critical role in shaping the performance of language models (Bostrom and Durrett, 2020; Conneau et al., 2020; Mielke et al., 2021; Gutierrez-Vasques et al., 2021; Rust et al., 2021; Ogueji et al., 2021; Maronikolakis et al., 2021; Oladipo et al., 2022; Liang et al., 2023; Ali et al., 2024; Limisiewicz et al., 2023; Petrov et al., 2023; Wang et al., 2024). Work in this area has shown that language-specific tokenizers can outperform multilingual ones (Rust et al., 2021). Moreover, incorporating morphological information into subword tokenization has been found to improve model performance across various languages (Hofmann et al., 2021; Alyafeai et al., 2023; Toraman et al., 2023; Fujii et al., 2023; Arnett et al., 2024).

When it comes to diacritization and its effect on tokenization, Gorman and Pinter (2025) point out that most text preprocessing pipelines used to train tokenizers do not follow consistent Unicode normalization and almost always strip diacritics. This practice leads to performance degradation in LLMs across various languages that use diacritics, including Arabic. Stripping diacritics causes them to be treated as out-of-vocabulary (OOV), preventing models from learning meaningful representations. Inconsistent usage of diacritics in the training data also leads to low-frequency diacritic tokens, causing over-segmentation (i.e., high token fertility) and increasing both computational and monetary costs. In our work, we take a systematic approach to quantify the performance of various LLMs across multiple benchmarks under varying degrees of diacritization.

LLM benchmarking. Several benchmarks have been proposed to evaluate LLMs on Arabic across a variety of tasks. Koto et al. (2024) introduced ArabicMMLU, an Arabic adaptation of the MMLU benchmark (Hendrycks et al., 2021), constructed from school exam questions. Mousi et al. (2025) presented AraDICE, a benchmark targeting dialectal Arabic and cultural understanding. Alwajih et al. (2025) developed Palm, which spans both Modern Standard Arabic (MSA) and dialects across 20 diverse topics. Hijazi et al. (2024) proposed ArabLegalEval to assess Arabic legal knowledge in LLMs. Ashraf et al. (2025) introduced a benchmark focused on evaluating LLM safety in Arabic, while Alghamdi et al. (2025) released AraTrust, designed to evaluate trustworthiness in Arabic LLMs. Finally, Sadallah et al. (2025) presented ArabCulture, a commonsense reasoning dataset in MSA that reflects cultural knowledge across 13 Arab countries.

Despite their breadth and value, these benchmarks overlook the role of diacritization in Arabic. Given that diacritics can significantly influence tokenization and model predictions, this remains a critical gap in the evaluation of Arabic LLMs. In this work, we address this limitation by systematically evaluating the performance of various LLMs on ArabicMMLU, AraTrust, and ArabCulture under varying degrees of diacritization.

Tokenizer	Vocabulary Size				#subwords / #words			#chars / #subwords		
	Total	Arabic	(%)	Diac.	`undiac`	`wild`	`full`	`undiac`	`wild`	`full`
AceGPT-v2 Instruct (8B)	128,256	1,176	0.9	77	2.6	4.2	5.8	1.8	1.5	1.5
ALLaM (7B)	64,000	36,797	57.5	954	1.4	2.7	4.0	3.2	2.4	2.2
Fanar Instruct (9B)	128,256	1,840	1.4	55	2.4	4.2	6.7	1.9	1.6	1.3
Jais Chat (13B)	84,992	8,518	10.0	7	1.9	5.0	9.3	2.4	1.3	0.9
Jais 2 Chat (8B)	150,222	9,390	6.3	5	1.9	5.5	10.1	2.4	1.2	0.9
Aya-Expanse (8B)	256,000	2,054	0.8	13	2.2	4.6	7.8	2.1	1.4	1.1
Llama-3.1 Instruct (8B)	128,256	1,176	0.9	77	2.6	4.2	5.8	1.8	1.5	1.5
Qwen3 (8B)	152,064	2,174	1.4	83	2.4	4.7	7.8	1.9	1.4	1.1
GPT-4o	200,000	2,127	1.1	92	2.4	4.1	6.3	1.9	1.6	1.4

Table 2: Tokenizer vocabulary statistics and corpus-level tokenization statistics on Wild2MaxDiacs (Elgamal et al., 2024). **Arabic** counts vocabulary items composed exclusively of Arabic characters; **(%)** is the corresponding share of the vocabulary. **Diac.** counts vocabulary items containing any Arabic diacritic. Subword ratios are reported for **`undiac`** (diacritics removed), **`wild`** (raw text), and **`full`** (full diacritization).

Tokenizer	%Words with Equal Subword Counts			Jaccard Similarity (Average)		
	`undiac-wild`	`undiac-full`	`wild-full`	`undiac-wild`	`undiac-full`	`wild-full`
AceGPT-v2 Instruct (8B)	8.6	0.2	24.3	0.43 $\pm$0.28	0.13 $\pm$0.12	0.40 $\pm$0.32
ALLaM (7B)	24.6	6.7	27.9	0.13 $\pm$0.21	0.01 $\pm$0.04	0.24 $\pm$0.34
Fanar Instruct (9B)	22.3	0.0	19.1	0.33 $\pm$0.26	0.10 $\pm$0.11	0.38 $\pm$0.33
Jais Chat (13B)	0.3	0.0	17.7	0.29 $\pm$0.28	0.05 $\pm$0.07	0.43 $\pm$0.32
Jais 2 Chat (8B)	0.1	0.0	17.8	0.27 $\pm$0.27	0.05 $\pm$0.07	0.46 $\pm$0.31
Aya-Expanse (8B)	0.3	0.0	18.2	0.37 $\pm$0.30	0.09 $\pm$0.12	0.45 $\pm$0.33
Llama-3.1 Instruct (8B)	8.6	0.2	24.3	0.43 $\pm$0.28	0.13 $\pm$0.12	0.40 $\pm$0.32
Qwen3 (8B)	7.4	0.0	18.0	0.37 $\pm$0.28	0.10 $\pm$0.11	0.44 $\pm$0.34
GPT-4o	22.6	0.0	20.7	0.32 $\pm$0.25	0.10 $\pm$0.11	0.37 $\pm$0.32

Table 3: Statistics on subword tokenization consistency computed on the Wild2MaxDiacs dataset. **%Words with Equal Subword Counts** is the percentage of words whose subword count remains unchanged between the diacritization settings. **Jaccard Similarity** is the overlap of subword sets between the diacritization settings.

4 Impact on Tokenization

4.1 Diacritics and Tokenization

Impact on subword length. Table 2 shows the tokenization statistics of the nine models we use in this study, measured on the Wild2MaxDiacs dataset (Elgamal et al., 2024) across three diacritization settings: **`undiac`**, **`wild`**, and **`full`**. We notice that the percentage of Arabic tokens in the vocabulary is comparable across most tokenizers, except for ALLaM, where Arabic tokens account for 57.5% of the vocabulary. Fertility (defined as the average number of subwords per word) consistently increases from the `undiac` to `wild` to `full` settings across all tokenizers. This indicates that the presence of diacritics, especially in fully diacritized form, leads to more fragmented tokenization, highlighting the limited diacritic awareness in the tokenizers. Token density (i.e., the average number of characters per subword) shows that higher levels of diacritization lead to tokenization behavior closer to the character level, most notably in Jais tokenizers, where a single diacritic character is tokenized into a multiple-byte sequence due to the missing diacritic entry in the vocabulary. See Section A for domain-wise statistics.

Tokenization consistency. Table 3 presents statistics on tokenization consistency across diacritization settings. We compare tokenizers using two metrics: the percentage of words whose subword count remains unchanged across paired diacritization conditions, and the average Jaccard similarity between the subword sets of those pairs. Among the models, ALLaM, Fanar, and GPT-4o show the highest percentage of equal subword counts in the `undiac-wild` condition, suggesting better robustness to wild diacritization. In contrast, models such as Jais and Aya exhibit poor tokenization stability across diacritization settings, indicating that these tokenizers are more sensitive to diacritics. It is worth noting that having close-matching subword counts does not necessarily imply similar tokenizations. This is reflected

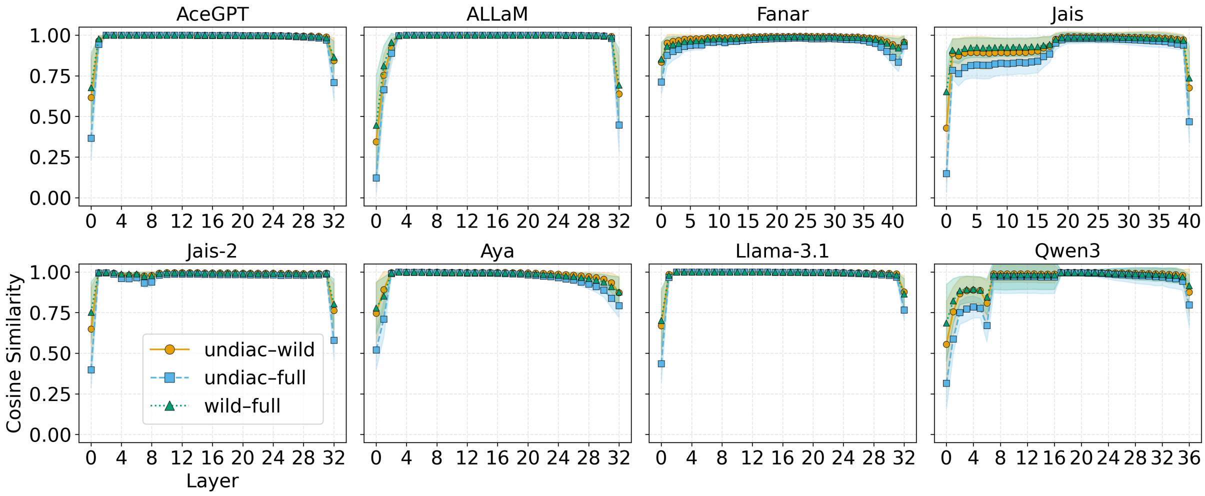

Figure 1: Layer-wise cosine similarities of word-level representations for `undiac-wild`, `undic-full`, and `wild-full` pairs. For each layer, we extract a single word vector by mean-pooling across its subword token representations. Cosine similarity values are then averaged over a 3,000-word subset of the Wild2MaxDiacs dataset.

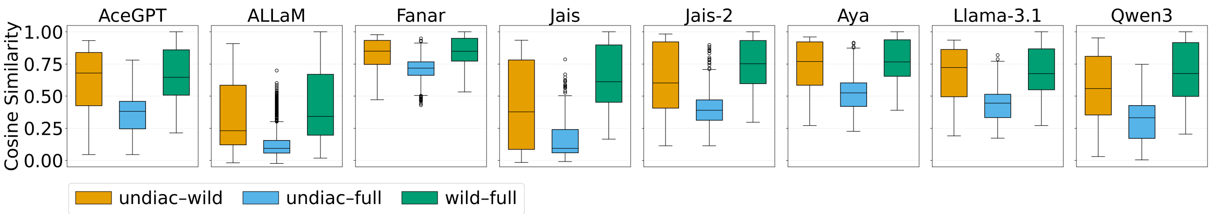

Figure 2: Cosine similarities of word-level representations from the first layer for `undiac-wild`, `undiac-full`, and `wild-full` parings. We provide domain-wise results in Section A.

in the relatively low Jaccard similarities observed even when the percentage of words with equal subword counts is high, as seen in ALLaM, Fanar, and GPT-4o. Meanwhile, high Jaccard similarity of `undiac-wild` in AceGPT and Llama models can be attributed to their near-character-level tokenization. See Section A for domain-wise statistics.

4.2 Diacritics and Internal Representation

Layer-wise cosine similarity. To assess the impact of diacritics beyond surface-level tokenization artifacts, we measure layer-wise cosine similarity between word-level representations across three diacritization settings. For each word in the Wild2MaxDiacs dataset, we compute the cosine similarity between its representations under different diacritization settings at each layer. Figure 1 shows how these similarities evolve across layers for each model. Across all models, we observe that cosine similarities increase across the lower layers and approach 1.0 in the middle layers. This can be attributed to inter-word anisotropy,[4] where representations become increasingly aligned in direction in intermediate layers (Ethayarajh, 2019; Godey et al., 2024). We also observe that the `undiac-full` pair exhibits the lowest cosine similarity, which reflects the lowest Jaccard similarity as seen in Table 3 (§4.1).

Embedding cosine similarity. To analyze the effect of diacritics at the embedding level, we present the distribution of cosine similarities between representations at layer 0 across different diacritization conditions in Figure 2. Across all models, the `undiac-full` comparison yields the lowest similarity scores, followed by `undiac-wild` and `wild-full`, reflecting a similar trend observed in surface-level tokenization differences. Among the models, Fanar exhibits higher cosine similarities across all diacritization conditions, suggesting that the embedding space is relatively invariant to diacritic presence, even in the case of `undiac-full`, where the percentage of words with equal subword counts is 0.0 with a low Jaccard similarity as seen in Table 3.

[4]See Section D for inter-word similarities within each diacritization setting.

5 Impact on Benchmark Tasks

5.1 Diacritic Removal and Restoration

To assess the impact of diacritics on benchmark task performance, we construct two modified versions of each dataset: one with diacritics removed (`undiac`) and another with automatically added diacritics (`full*`). For diacritic removal, we use the `dediac_ar` function in CAMeL Tools (Obeid et al., 2020). For diacritic restoration, we use an automatic diacritization system based on a BERT-based morphological disambiguation model (Inoue et al., 2022). Specifically, we use the extended version introduced by Elgamal et al. (2024), which incorporates an improved re-ranking strategy, an expanded morphological analyzer database, and full contextual post-editing of diacritics, achieving a word-level diacritization accuracy of 88.9%. As a result, we obtain three versions of each sample in a given dataset (`undiac`, `wild`, and `full*`) that differ only in the degree of diacritization.

5.2 Benchmark

We evaluate on three widely used Arabic LLM benchmarks: ArabicMMLU (Koto et al., 2024), ArabCulture (Sadallah et al., 2025), and AraTrust (Alghamdi et al., 2025). Diacritic processing, as described above, is applied to the question and answer option texts, as well as to the context field in ArabicMMLU when available. Table 4 shows the statistics of diacritic presence across these datasets under two conditions: the original version (`wild`) and the automatically diacritized version (`full*`). All benchmarks exhibit a similar percentage of lines with diacritics (∼16-17%) in the wild setting. However, ArabicMMLU contains substantially more diacritized words (19.0%) compared to ArabCulture (1.5%) and AraTrust (1.2%). The `full*` setting results in near-complete diacritization and is most comparable in coverage to the Children domain in the PDD dataset. We also measure the similarity between the diacritic type distributions of the benchmark datasets and the PDD subsets using Jensen-Shannon divergence (Lin, 1991). We observe that ArabicMMLU is most similar to the News subset in terms of diacritic type distribution in the `wild` setting, while ArabCulture and AraTrust are closer to the Children subset. The `full` version is the closest to the News domain across datasets. For details, see Section E.

Benchmark	Diac.	%lines (w/ diac)	%words (w/ diac)	#diacs (per word)
ArabicMMLU	`wild`	16.9	19.0	3.1
	`full*`	99.8	97.5	3.6
ArabCulture	`wild`	17.2	1.5	1.0
	`full*`	100.0	97.9	3.6
AraTrust	`wild`	16.1	1.2	1.0
	`full*`	100.0	97.3	3.5

Table 4: Statistics of diacritic usage in three Arabic LLM benchmarks under `wild` and `full*` diacritization levels (**Diac.**).

6 Experiment

6.1 Settings

Models. We run zero-shot experiments across nine models categorized into three groups: (a) Arabic-centric open-weight models, including AceGPT-v2-8B (Liang et al., 2024), ALLaM-7B (Bari et al., 2024), Fanar-9B (Fanar Team et al., 2025), Jais-13B, and Jais-2-8B (Sengupta et al., 2023); (b) multi-lingual open-weight models, including Aya-Expanse-8B (Dang et al., 2024), Llama-3.1-8B (Grattafiori et al., 2024), and Qwen3-8B (Yang et al., 2025); and (c) a closed-weight model, GPT-4o (OpenAI et al., 2024). Based on initial experiments, we found that instruction-tuned versions of the open-weight models performed better than their base counterparts, and we use them throughout this work. For GPT-4o, we use the snapshot of `gpt-4o-2024-08-06`. See Section B for computational budgets to run these models.

Prompting. Following previous studies (Koto et al., 2024; Sadallah et al., 2025; Alghamdi et al., 2025), we adopt the best-performing setup, which uses English prompts with English alphabetical output for each benchmark dataset. For open-weight models, we run our experiments on the LM Evaluation Harness (Gao et al., 2024), specifying zero temperature (greedy sampling) following Sadallah et al. (2025). We follow the standard evaluation strategy of multiple-choice questions in the LM Evaluation Harness, where the model computes the likelihood of the prompt concatenated with each answer option, and selects the option with the highest likelihood as its prediction. For GPT-4o, we use OpenAI's Batch API, specifying a JSON response containing only the answer character, with temperature set to 0 and a fixed random seed of 12345 to ensure determinism. See Section C for further details on prompts.

Model	ArabicMMLU ($n = 14,455$)			ArabCulture ($n = 3,482$)			AraTrust ($n = 522$)			Macro Average		
	`undiac`	`wild`	`full*`	`undiac`	`wild`	`full*`	`undiac`	`wild`	`full*`	`undiac`	`wild`	`full*`
AceGPT-v2 Chat (8B)	57.2†	57.0	46.1†	35.3	35.4	34.4	68.8	69.2	58.8†	53.8	53.9	46.4
ALLaM (7B)	70.0	70.0	64.2†	37.1	37.2	35.3†	83.3	83.5	77.0†	63.5	63.6	58.9
Fanar Instruct (9B)	65.3	65.3	55.6†	44.7†	44.9	42.4†	85.4	85.4	78.0†	65.1	65.2	58.6
Jais Chat (13B)	58.2	58.1	46.3†	42.3†	42.7	40.0†	78.7	79.5	56.5†	59.8	60.1	47.6
Jais 2 Chat (8B)	58.6	58.7	50.9†	36.9	36.8	37.0	73.6	73.6	63.4†	56.4	56.4	50.4
Aya-Expanse (8B)	58.9†	59.2	50.7†	35.0	34.9	32.9†	85.6	85.4	76.4†	59.8	59.9	53.3
Llama-3.1 Instruct (8B)	56.2†	56.4	42.5†	35.4	35.3	33.4†	83.3	83.5	62.1†	58.3	58.4	46.0
Qwen3 (8B)	62.5	62.4	56.6†	35.0	35.0	33.3†	85.1	84.7	83.1	60.9	60.7	57.7
GPT-4o	**80.7**	**80.8**	**79.4**†	**88.2**	**88.3**	**87.2**†	**92.9**	**92.5**	**92.9**	**87.3**	**87.2**	**86.5**
Model Average	63.1	63.1	54.7	43.3	43.4	41.8	81.9	81.9	72.0	62.7	62.8	56.2

Table 5: Accuracy scores of LLMs on benchmark tasks (ArabicMMLU, ArabCulture, AraTrust) across three diacritization settings (`undiac`, `wild`, and `full*`). † indicates a statistically significant difference (McNemar's test, $p < 0.05$) from the `wild` setting. The best-performing scores in each setting are highlighted in bold. Underlined scores are the best scores within the open-weight models.

Model	ArabicMMLU* ($n = 2,437$)			ArabCulture* ($n = 599$)			AraTrust* ($n = 84$)			Macro Average		
	`undiac`	`wild`	`full*`	`undiac`	`wild`	`full*`	`undiac`	`wild`	`full*`	`undiac`	`wild`	`full*`
AceGPT-v2 Chat (8B)	57.7†	56.7	45.0†	36.2	37.1	32.6†	78.6	81.0	72.6	57.5	58.2	50.0
ALLaM (7B)	73.3	73.7	68.7†	35.7	36.1	34.7	91.7	92.9	85.7	66.9	67.5	63.1
Fanar Instruct (9B)	68.9	69.1	57.9†	60.8†	61.9	55.8†	86.9	86.9	79.8	72.2	72.7	64.5
Jais Chat (13B)	59.2	58.7	44.3†	51.6†	53.6	43.9†	77.4	82.1	56.0†	62.7	64.8	48.1
Jais 2 Chat (8B)	62.9	63.5	51.4†	47.9	47.2	45.6	81.0	81.0	69.0†	63.9	63.9	55.3
Aya-Expanse (8B)	60.5†	62.3	51.0†	37.9	37.6	32.6†	94.0	92.9	86.9	64.2	64.2	56.8
Llama-3.1 Instruct (8B)	57.9†	59.1	43.3†	37.2	36.9	32.7†	91.7	92.9	65.5†	62.3	62.9	47.4
Qwen3 (8B)	64.3	63.9	57.1†	38.6	38.7	33.9†	86.9	84.5	83.3	63.2	62.4	58.1
GPT-4o	**82.4**	**82.9**	**80.5**†	**92.5**	**92.5**	**92.7**	**96.4**	**95.2**	**95.2**	**90.5**	**90.2**	**89.5**
Model Average	65.2	65.5	55.5	48.7	49.1	44.9	87.2	87.7	77.1	67.0	67.4	59.2

Table 6: Accuracy scores on benchmark subsets containing at least one diacritic in the Arabic text.

6.2 Results and Analysis

Main results. Table 5 shows the performance of the nine models across the three diacritization settings (`undiac`, `wild`, and `full*`) on the benchmark datasets (ArabicMMLU, ArabCulture, and AraTrust). Among all the models, GPT-4o consistently outperforms all others, demonstrating superior performance and robustness to the diacritic perturbations. Among open-weight models, ALLaM performs best on ArabicMMLU, Fanar on ArabCulture, and Aya on AraTrust across the three diacritization settings. For most models and benchmarks, removing diacritics (`undiac`) resulted in performance differences that are small and not statistically significant (McNemar's test, $p < 0.05$) relative to the original `wild` setting. Exceptions occur in five cases, where removing diacritics yields a statistically significant performance drop for Aya and Llama on ArabicMMLU, and for Fanar and Jais Chat (13B) on ArabCulture, while it yields a significant performance improvement for AceGPT on ArabicMMLU.

On the other hand, automatic diacritization consistently degrades performance across models and benchmarks, with statistically significant decreases observed in most cases, except for AceGPT and Jais-2 on ArabCulture, and for Qwen and GPT-4o on AraTrust. Overall, these results suggest that while modern LLMs are robust to the limited diacritics naturally found in typical real-world texts, they struggle with extensively diacritized input. This finding is particularly relevant for domains such as children's literature, educational materials, and religious texts, where diacritics are commonly used, highlighting the importance of careful diacritic handling when developing Arabic LLMs.

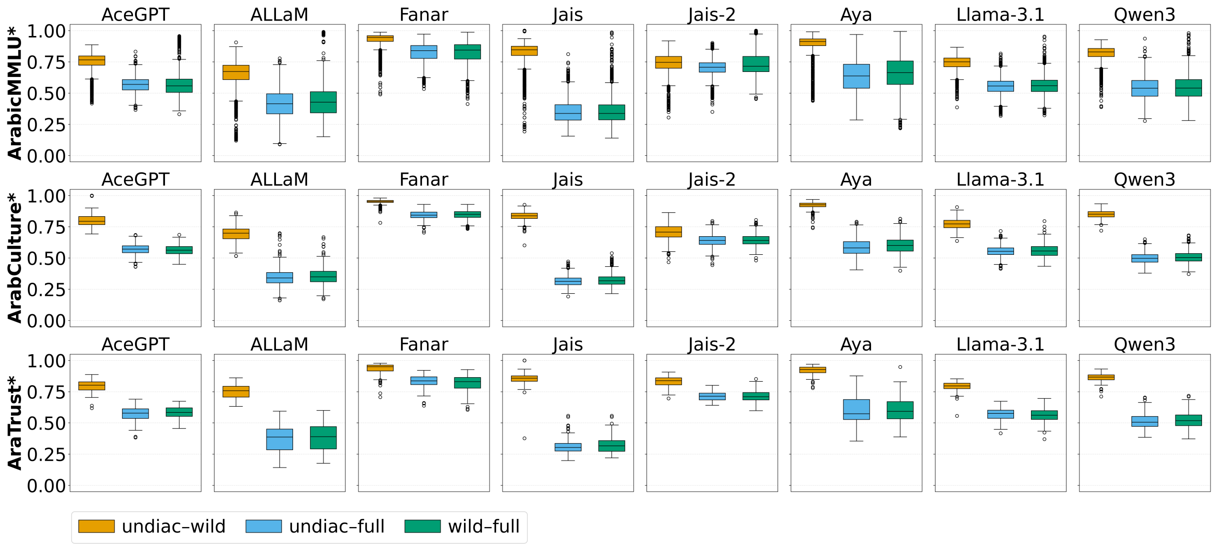

Figure 3: Cosine similarities of sentence-level representations from the first layer for `undiac-wild`, `undiac-full`, and `wild-full` parings. Sentence embeddings are obtained by mean-pooling token representations at the first layer. Each boxplot summarizes similarity scores across sentence pairs per benchmark (rows) and model (columns). Higher similarity indicates less representational change induced by diacritization.

Subsets with wild diacritics. In Table 6, we report on subsets of the benchmarks containing at least one Arabic diacritic in the original text, in order to explicitly examine model sensitivity to diacritics. Consistent with the overall results (Table 5), diacritic removal has minimal impact, whereas full diacritization degrades performance. Among Arabic-centric models, Jais Chat (13B) is the most sensitive to diacritics, exhibiting a substantial performance drop from `wild` to the `full*` setting across all benchmarks: 14.4 points decrease on ArabicMMLU, 9.7 on ArabCulture, and 26.2 on AraTrust. This can be attributed to the tokenization instability previously observed in the Jais tokenizer (§4.1), where the presence of diacritics leads to substantial subword fragmentation. Finally, we note that retaining wild diacritics often yields better average performance compared to removing them altogether. This finding contrasts with a historically common preprocessing practice in Arabic NLP of stripping diacritics during preprocessing (Antoun et al., 2020; Inoue et al., 2021), suggesting that such normalization may not be necessary and that diacritics may instead need to be explicitly modeled to robustly handle text with diacritics.

Internal representation. Figure 3 shows cosine similarities of sentence-level representations at the first layer for three diacritization pairs (`undiac-wild`, `undiac-full`, `wild-full`), across three benchmarks (rows) and eight open-weight models (columns). Across all benchmarks and models, the `undiac-wild` pairs consistently exhibit the highest cosine similarity, suggesting minimal representational change between these two settings. This aligns with the minimal performance variation observed between the `undiac` and `wild` conditions. Conversely, the `undiac-full` and `wild-full` comparisons show lower similarity scores, indicating significant representational shifts due to full diacritization, aligning with its observed performance degradation under full diacritization.

7 Conclusion and Future Work

In this study, we systematically analyzed the impact of Arabic diacritics on tokenization and the performance of a range of LLMs across three standard Arabic LLM benchmarks. Our findings show that while modern LLMs exhibit reasonable robustness to naturally occurring diacritics, a high degree of diacritization leads to substantially increased token fragmentation and performance degradation. These findings highlight the need for thoughtful diacritic processing and tokenizer configuration in building LLMs for Arabic.

For future work, we plan to explore simulating domain-specific diacritic usage distributions to evaluate LLMs under a wider range of diacritization conditions. We also aim to investigate strategies for optimizing the degree of diacritization. In addition, we plan to extend our evaluation beyond multiple-choice benchmarks.

Limitations

This study has several limitations. First, our evaluation is limited to zero-shot settings, potentially differing from performance in few-shot or fine-tuned scenarios. Second, automatic diacritization inevitably introduces errors, which could influence the observed negative impact on model performance. Third, due to resource constraints, our experiments cover a limited selection of models and benchmarks; additional models or tasks may yield differing insights. Finally, our analysis primarily targets Modern Standard Arabic, leaving open questions regarding the impact of diacritics in dialectal Arabic and other morphologically rich languages.

Ethical Considerations

We use publicly available datasets and language models solely for the purpose of evaluating language model behavior under varying degrees of diacritization. We do not anticipate any potential risks associated with this work, as it does not involve the collection of personal data, sensitive content, or human subjects. We used AI writing assistance within the scope of "Assistance purely with the language of the paper" described in the ACL Policy on Publication Ethics.

Acknowledgement

We acknowledge the support of the High Performance Computing Center at New York University Abu Dhabi. We thank the anonymous reviewers for their insightful and constructive comments. We also thank Salam Khalifa, Tatsuki Kuribayashi, and Tatsuya Hiraoka for their helpful discussions and valuable feedback.

References

Gheith A. Abandah, Alex Graves, Balkees Al-Shagoor, Alaa Arabiyat, Fuad Jamour, and Majid Al-Taee. 2015. Automatic diacritization of Arabic text using recurrent neural networks. *International Journal on Document Analysis and Recognition (IJDAR)*, 18(2):183–197.

Hanan Aldarmaki and Ahmad Ghannam. 2023. Diacritic Recognition Performance in Arabic ASR. In *Proc. INTERSPEECH 2023*, pages 361–365.

Emad A. Alghamdi, Reem Masoud, Deema Alnuhait, Afnan Y. Alomairi, Ahmed Ashraf, and Mohamed Zaytoon. 2025. AraTrust: An evaluation of trustworthiness for LLMs in Arabic. In *Proceedings of the 31st International Conference on Computational Linguistics*, pages 8664–8679, Abu Dhabi, UAE. Association for Computational Linguistics.

Mehdi Ali, Michael Fromm, Klaudia Thellmann, Richard Rutmann, Max Lübbering, Johannes Leveling, Katrin Klug, Jan Ebert, Niclas Doll, Jasper Schulze Buschhoff, Charvi Jain, Alexander Arno Weber, Lena Jurkschat, Hammam Abdelwahab, Chelsea John, Pedro Ortiz Suarez, Malte Ostendorff, Samuel Weinbach, Rafet Sifa, and 2 others. 2024. Tokenizer choice for llm training: Negligible or crucial? *Preprint*, arXiv:2310.08754.

Badr AlKhamissi, Muhammad ElNokrashy, and Mohamed Gabr. 2020. Deep diacritization: Efficient hierarchical recurrence for improved Arabic diacritization. In *Proceedings of the Fifth Arabic Natural Language Processing Workshop*, pages 38–48, Barcelona, Spain (Online). Association for Computational Linguistics.

Sawsan Alqahtani, Hanan Aldarmaki, and Mona Diab. 2019. Homograph disambiguation through selective diacritic restoration. In *Proceedings of the Fourth Arabic Natural Language Processing Workshop*, pages 49–59, Florence, Italy. Association for Computational Linguistics.

Sawsan Alqahtani, Mahmoud Ghoneim, and Mona Diab. 2016. Investigating the impact of various partial diacritization schemes on Arabic-English statistical machine translation. In *Conferences of the Association for Machine Translation in the Americas: MT Researchers' Track*, pages 191–204, Austin, TX, USA. The Association for Machine Translation in the Americas.

Fakhraddin Alwajih, Abdellah El Mekki, Samar Mohamed Magdy, AbdelRahim A. Elmadany, Omer Nacar, El Moatez Billah Nagoudi, Reem Abdel-Salam, Hanin Atwany, Youssef Nafea, Abdulfattah Mohammed Yahya, Rahaf Alhamouri, Hamzah A. Alsayadi, Hiba Zayed, Sara Shatnawi, Serry Sibaee, Yasir Ech-chammakhy, Walid Al-Dhabyani, Marwa Mohamed Ali, Imen Jarraya, and 25 others. 2025. Palm: A culturally inclusive and linguistically diverse dataset for Arabic LLMs. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 32871–32894, Vienna, Austria. Association for Computational Linguistics.

Zaid Alyafeai, Maged S Al-shaibani, Mustafa Ghaleb, and Irfan Ahmad. 2023. Evaluating various tokenizers for Arabic text classification. *Neural Processing Letters*, 55(3):2911–2933.

Wissam Antoun, Fady Baly, and Hazem Hajj. 2020. AraBERT: Transformer-based model for Arabic language understanding. In *Proceedings of the 4th Workshop on Open-Source Arabic Corpora and Processing Tools, with a Shared Task on Offensive Language Detection*, pages 9–15, Marseille, France. European Language Resource Association.

Catherine Arnett, Pamela D. Rivière, Tyler A. Chang, and Sean Trott. 2024. Different tokenization schemes lead to comparable performance in Spanish number agreement. *Preprint*, arXiv:2403.13754.

Yasser Ashraf, Yuxia Wang, Bin Gu, Preslav Nakov, and Timothy Baldwin. 2025. Arabic dataset for LLM safeguard evaluation. 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 5529–5546, Albuquerque, New Mexico. Association for Computational Linguistics.

Parnia Bahar, Mattia Di Gangi, Nick Rossenbach, and Mohammad Zeineldeen. 2023. Take the Hint: Improving Arabic Diacritization with Partially-Diacritized Text. In *Proc. INTERSPEECH 2023*, pages 3949–3953.

M Saiful Bari, Yazeed Alnumay, Norah A. Alzahrani, Nouf M. Alotaibi, Hisham A. Alyahya, Sultan Al-Rashed, Faisal A. Mirza, Shaykhah Z. Alsubaie, Hassan A. Alahmed, Ghadah Alabduljabbar, Raghad Alkhathran, Yousef Almushayqih, Raneem Alnajim, Salman Alsubaihi, Maryam Al Mansour, Majed Alrubaian, Ali Alammari, Zaki Alawami, Abdulmohsen Al-Thubaity, and 6 others. 2024. ALLaM: Large language models for Arabic and English. *Preprint*, arXiv:2407.15390.

Yonatan Belinkov and James Glass. 2015. Arabic diacritization with recurrent neural networks. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 2281–2285, Lisbon, Portugal. Association for Computational Linguistics.

Kaj Bostrom and Greg Durrett. 2020. Byte pair encoding is suboptimal for language model pretraining. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4617–4624, Online. Association for Computational Linguistics.

Wei-Rui Chen, Ife Adebara, and Muhammad Abdul-Mageed. 2024. Interplay of machine translation, diacritics, and diacritization. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 7559–7601, Mexico City, Mexico. Association for Computational Linguistics.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised cross-lingual representation learning at scale. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8440–8451, Online. Association for Computational Linguistics.

John Dang, Shivalika Singh, Daniel D'souza, Arash Ahmadian, Alejandro Salamanca, Madeline Smith, Aidan Peppin, Sungjin Hong, Manoj Govindassamy, Terrence Zhao, Sandra Kublik, Meor Amer, Viraat Aryabumi, Jon Ander Campos, Yi-Chern Tan, Tom Kocmi, Florian Strub, Nathan Grinsztajn, Yannis Flet-Berliac, and 26 others. 2024. Aya Expanse: Combining research breakthroughs for a new multilingual frontier. *Preprint*, arXiv:2412.04261.

Kareem Darwish, Ahmed Abdelali, Hamdy Mubarak, and Mohamed Eldesouki. 2021. Arabic diacritic recovery using a feature-rich bilstm model. *ACM Trans. Asian Low-Resour. Lang. Inf. Process.*, 20(2).

Kareem Darwish, Hamdy Mubarak, and Ahmed Abdelali. 2017. Arabic diacritization: Stats, rules, and hacks. In *Proceedings of the Third Arabic Natural Language Processing Workshop*, pages 9–17, Valencia, Spain. Association for Computational Linguistics.

Fathi Debili and Hadhemi Achour. 1998. Voyellation automatique de l'arabe. In *Computational Approaches to Semitic Languages*.

Mona Diab, Mahmoud Ghoneim, and Nizar Habash. 2007. Arabic diacritization in the context of statistical machine translation. In *Proceedings of Machine Translation Summit XI: Papers*, Copenhagen, Denmark.

Y. A. El-Imam. 2004. Phonetization of Arabic: Rules and Algorithms. In *Computer Speech and Language (CSL)*, pages 339–373.

Salman Elgamal, Ossama Obeid, Mhd Kabbani, Go Inoue, and Nizar Habash. 2024. Arabic diacritics in the wild: Exploiting opportunities for improved diacritization. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 14815–14829, Bangkok, Thailand. Association for Computational Linguistics.

Muhammad Morsy Elmallah, Mahmoud Reda, Kareem Darwish, Abdelrahman El-Sheikh, Ashraf Hatim Elneima, Murtadha Aljubran, Nouf Alsaeed, Reem Mohammed, and Mohamed Al-Badrashiny. 2024. Arabic diacritization using morphologically informed character-level model. In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, pages 1446–1454, Torino, Italia. ELRA and ICCL.

Muhammad ElNokrashy and Badr AlKhamissi. 2024. Partial diacritization: A context-contrastive inference approach. *Preprint*, arXiv:2401.08919.

Saeed Esmail, Kfir Bar, and Nachum Dershowitz. 2022. How much does lookahead matter for disambiguation? partial Arabic diacritization case study. *Computational Linguistics*, 48(4):1103–1123.

Kawin Ethayarajh. 2019. How contextual are contextualized word representations? Comparing the geometry of BERT, ELMo, and GPT-2 embeddings. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 55–65, Hong Kong, China. Association for Computational Linguistics.

Ali Fadel, Ibraheem Tuffaha, Bara' Al-Jawarneh, and Mahmoud Al-Ayyoub. 2019. Neural Arabic text diacritization: State of the art results and a novel approach for machine translation. In *Proceedings of the*

6th Workshop on Asian Translation, pages 215–225, Hong Kong, China. Association for Computational Linguistics.

Fanar Team, Ummar Abbas, Mohammad Shahmeer Ahmad, Firoj Alam, Enes Altinisik, Ehsannedin Asgari, Yazan Boshmaf, Sabri Boughorbel, Sanjay Chawla, Shammur Chowdhury, Fahim Dalvi, Kareem Darwish, Nadir Durrani, Mohamed Elfeky, Ahmed Elmagarmid, Mohamed Eltabakh, Masoomali Fatehkia, Anastasios Fragkopoulos, Maram Hasanain, and 23 others. 2025. Fanar: An Arabic-centric multimodal generative AI platform.

Takuro Fujii, Koki Shibata, Atsuki Yamaguchi, Terufumi Morishita, and Yasuhiro Sogawa. 2023. How do different tokenizers perform on downstream tasks in scriptio continua languages?: A case study in Japanese. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 4: Student Research Workshop)*, pages 39–49, Toronto, Canada. Association for Computational Linguistics.

Leo Gao, Jonathan Tow, Baber Abbasi, Stella Biderman, Sid Black, Anthony DiPofi, Charles Foster, Laurence Golding, Jeffrey Hsu, Alain Le Noac'h, Haonan Li, Kyle McDonell, Niklas Muennighoff, Chris Ociepa, Jason Phang, Laria Reynolds, Hailey Schoelkopf, Aviya Skowron, Lintang Sutawika, and 5 others. 2024. The language model evaluation harness.

Nathan Godey, Éric Clergerie, and Benoît Sagot. 2024. Anisotropy is inherent to self-attention in transformers. In *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 35–48, St. Julian's, Malta. Association for Computational Linguistics.

Kyle Gorman and Yuval Pinter. 2025. Don't touch my diacritics. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 2: Short Papers)*, pages 285–291, Albuquerque, New Mexico. Association for Computational Linguistics.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, Arthur Hinsvark, and 542 others. 2024. The Llama 3 herd of models. *Preprint*, arXiv:2407.21783.

Ximena Gutierrez-Vasques, Christian Bentz, Olga Sozinova, and Tanja Samardzic. 2021. From characters to words: the turning point of BPE merges. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 3454–3468, Online. Association for Computational Linguistics.

Nizar Habash and Owen Rambow. 2007. Arabic diacritization through full morphological tagging. In *Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics; Companion Volume, Short Papers*, pages 53–56, Rochester, New York. Association for Computational Linguistics.

Nizar Habash, Anas Shahrour, and Muhamed Al-Khalil. 2016. Exploiting Arabic diacritization for high quality automatic annotation. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 4298–4304, Portorož, Slovenia. European Language Resources Association (ELRA).

Nizar Habash, Abdelhadi Soudi, and Tim Buckwalter. 2007. On Arabic Transliteration. In A. van den Bosch and A. Soudi, editors, *Arabic Computational Morphology: Knowledge-based and Empirical Methods*, pages 15–22. Springer, Netherlands.

Nizar Y Habash. 2010. *Introduction to Arabic natural language processing*, volume 3. Morgan & Claypool Publishers.

Osama Hamed and Torsten Zesch. 2018. The role of diacritics in increasing the difficulty of Arabic lexical recognition tests. In *Proceedings of the 7th workshop on NLP for Computer Assisted Language Learning*, pages 23–31, Stockholm, Sweden. LiU Electronic Press.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2021. Measuring massive multitask language understanding. In *International Conference on Learning Representations*.

Faris Hijazi, Somayah Alharbi, Abdulaziz AlHussein, Harethah Shairah, Reem Alzahrani, Hebah Alshamlan, George Turkiyyah, and Omar Knio. 2024. ArabLegalEval: A multitask benchmark for assessing Arabic legal knowledge in large language models. In *Proceedings of the Second Arabic Natural Language Processing Conference*, pages 225–249, Bangkok, Thailand. Association for Computational Linguistics.

Valentin Hofmann, Janet Pierrehumbert, and Hinrich Schütze. 2021. Superbizarre is not superb: Derivational morphology improves BERT's interpretation of complex words. 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 3594–3608, Online. Association for Computational Linguistics.

Go Inoue, Bashar Alhafni, Nurpeiis Baimukan, Houda Bouamor, and Nizar Habash. 2021. The interplay of variant, size, and task type in Arabic pre-trained language models. In *Proceedings of the Sixth Arabic Natural Language Processing Workshop*, pages 92–104, Kyiv, Ukraine (Virtual). Association for Computational Linguistics.

Go Inoue, Salam Khalifa, and Nizar Habash. 2022. Morphosyntactic tagging with pre-trained language models for Arabic and its dialects. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 1708–1719, Dublin, Ireland. Association for Computational Linguistics.

Fajri Koto, Haonan Li, Sara Shatnawi, Jad Doughman, Abdelrahman Sadallah, Aisha Alraeesi, Khalid Almubarak, Zaid Alyafeai, Neha Sengupta, Shady Shehata, Nizar Habash, Preslav Nakov, and Timothy Baldwin. 2024. ArabicMMLU: Assessing massive multitask language understanding in Arabic. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 5622–5640, Bangkok, Thailand. Association for Computational Linguistics.

Davis Liang, Hila Gonen, Yuning Mao, Rui Hou, Naman Goyal, Marjan Ghazvininejad, Luke Zettlemoyer, and Madian Khabsa. 2023. XLM-V: Overcoming the vocabulary bottleneck in multilingual masked language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 13142–13152, Singapore. Association for Computational Linguistics.

Juhao Liang, Zhenyang Cai, Jianqing Zhu, Huang Huang, Kewei Zong, Bang An, Mosen Alharthi, Juncai He, Lian Zhang, Haizhou Li, Benyou Wang, and Jinchao Xu. 2024. Alignment at pre-training! towards native alignment for arabic LLMs. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*.

Tomasz Limisiewicz, Jiří Balhar, and David Mareček. 2023. Tokenization impacts multilingual language modeling: Assessing vocabulary allocation and overlap across languages. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 5661–5681, Toronto, Canada. Association for Computational Linguistics.

Jianhua Lin. 1991. Divergence measures based on the shannon entropy. *IEEE Transactions on Information Theory*, 37(1):145–151.

Antonis Maronikolakis, Philipp Dufter, and Hinrich Schütze. 2021. Wine is not v i n. on the compatibility of tokenizations across languages. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 2382–2399, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Sabrina J. Mielke, Zaid Alyafeai, Elizabeth Salesky, Colin Raffel, Manan Dey, Matthias Gallé, Arun Raja, Chenglei Si, Wilson Y. Lee, Benoît Sagot, and Samson Tan. 2021. Between words and characters: A brief history of open-vocabulary modeling and tokenization in NLP. *Preprint*, arXiv:2112.10508.

Abubakr Mohamed and Hamdy Mubarak. 2025. Advancing Arabic diacritization: Improved datasets, benchmarking, and state-of-the-art models. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 16707–16719, Suzhou, China. Association for Computational Linguistics.

Basel Mousi, Nadir Durrani, Fatema Ahmad, Md. Arid Hasan, Maram Hasanain, Tameem Kabbani, Fahim Dalvi, Shammur Absar Chowdhury, and Firoj Alam. 2025. AraDiCE: Benchmarks for dialectal and cultural capabilities in LLMs. In *Proceedings of the 31st International Conference on Computational Linguistics*, pages 4186–4218, Abu Dhabi, UAE. Association for Computational Linguistics.

Hamdy Mubarak, Ahmed Abdelali, Hassan Sajjad, Younes Samih, and Kareem Darwish. 2019. Highly effective Arabic diacritization using sequence to sequence modeling. 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 2390–2395, Minneapolis, Minnesota. Association for Computational Linguistics.

Rani Nelken and Stuart M. Shieber. 2005. Arabic diacritization using weighted finite-state transducers. In *Proceedings of the ACL Workshop on Computational Approaches to Semitic Languages*, pages 79–86, Ann Arbor, Michigan. Association for Computational Linguistics.

Ossama Obeid, Nasser Zalmout, Salam Khalifa, Dima Taji, Mai Oudah, Bashar Alhafni, Go Inoue, Fadhl Eryani, Alexander Erdmann, and Nizar Habash. 2020. CAMeL tools: An open source python toolkit for Arabic natural language processing. In *Proceedings of the Twelfth Language Resources and Evaluation Conference*, pages 7022–7032, Marseille, France. European Language Resources Association.

Kelechi Ogueji, Yuxin Zhu, and Jimmy Lin. 2021. Small data? no problem! exploring the viability of pretrained multilingual language models for low-resourced languages. In *Proceedings of the 1st Workshop on Multilingual Representation Learning*, pages 116–126, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Akintunde Oladipo, Odunayo Ogundepo, Kelechi Ogueji, and Jimmy Lin. 2022. An exploration of vocabulary size and transfer effects in multilingual language models for african languages. In *3rd Workshop on African Natural Language Processing*.

OpenAI, Aaron Hurst, Adam Lerer, Adam P. Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, AJ Ostrow, Akila Welihinda, Alan Hayes, Alec Radford, Aleksander Mądry, Alex Baker-Whitcomb, Alex Beutel, Alex Borzunov, Alex Carney, Alex Chow, Alex Kirillov, Alex Nichol, and 400 others. 2024. GPT-4o system card. *Preprint*, arXiv:2410.21276.

Arfath Pasha, Mohamed Al-Badrashiny, Mona Diab, Ahmed El Kholy, Ramy Eskander, Nizar Habash, Manoj Pooleery, Owen Rambow, and Ryan Roth. 2014. MADAMIRA: A fast, comprehensive tool for morphological analysis and disambiguation of Arabic. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*, pages 1094–1101, Reykjavik, Iceland. European Language Resources Association (ELRA).

Aleksandar Petrov, Emanuele La Malfa, Philip H. S. Torr, and Adel Bibi. 2023. Language model tokenizers introduce unfairness between languages. In *Advances in Neural Information Processing Systems*.

Mohsen A. A. Rashwan, Ahmad A. Al Sallab, Hazem M. Raafat, and Ahmed Rafea. 2015. Deep learning framework with confused sub-set resolution architecture for automatic arabic diacritization. *IEEE/ACM*

Transactions on Audio, Speech, and Language Processing, 23(3):505–516.

Ryan Roth, Owen Rambow, Nizar Habash, Mona Diab, and Cynthia Rudin. 2008. Arabic morphological tagging, diacritization, and lemmatization using lexeme models and feature ranking. In *Proceedings of ACL-08: HLT, Short Papers*, pages 117–120, Columbus, Ohio. Association for Computational Linguistics.

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.

Abdelrahman Sadallah, Junior Cedric Tonga, Khalid Almubarak, Saeed Almheiri, Farah Atif, Chatrine Qwaider, Karima Kadaoui, Sara Shatnawi, Yaser Alesh, and Fajri Koto. 2025. Commonsense reasoning in Arab culture. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 7695–7710, Vienna, Austria. Association for Computational Linguistics.

Neha Sengupta, Sunil Kumar Sahu, Bokang Jia, Satheesh Katipomu, Haonan Li, Fajri Koto, William Marshall, Gurpreet Gosal, Cynthia Liu, Zhiming Chen, Osama Mohammed Afzal, Samta Kamboj, Onkar Pandit, Rahul Pal, Lalit Pradhan, Zain Muhammad Mujahid, Massa Baali, Xudong Han, Sondos Mahmoud Bsharat, and 13 others. 2023. Jais and Jais-chat: Arabic-centric foundation and instruction-tuned open generative large language models. *Preprint*, arXiv:2308.16149.

Cagri Toraman, Eyup Halit Yilmaz, Furkan Şahinuç, and Oguzhan Ozcelik. 2023. Impact of tokenization on language models: An analysis for Turkish. *ACM Trans. Asian Low-Resour. Lang. Inf. Process.*, 22(4).

Catalin Ungurean, Dragos Burileanu, Vladimir Popescu, Cristian Negrescu, and Aurelian Dervis. 2008. Automatic diacritic restoration for a TTS-based e-mail reader application. *UPB Scientific Bulletin, Series C*, 70(4):3–12.

Dixuan Wang, Yanda Li, Junyuan Jiang, Zepeng Ding, Guochao Jiang, Jiaqing Liang, and Deqing Yang. 2024. Tokenization matters! degrading large language models through challenging their tokenization. *Preprint*, arXiv:2405.17067.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, and 41 others. 2025. Qwen3 technical report. *Preprint*, arXiv:2505.09388.

Nasser Zalmout and Nizar Habash. 2020. Joint diacritization, lemmatization, normalization, and fine-grained morphological tagging. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8297–8307, Online. Association for Computational Linguistics.

Imed Zitouni, Jeffrey S. Sorensen, and Ruhi Sarikaya. 2006. Maximum entropy based restoration of Arabic diacritics. In *Proceedings of the International Conference on Computational Linguistics and the Conference of the Association for Computational Linguistics (COLING-ACL)*, pages 577–584, Sydney, Australia.

A Domain-wise Statistics

We provide domain-wise statistics in Figures 4 to 7. Figure 4 shows the average subword length for each domain in the Wild2MaxDiacs dataset (Elgamal et al., 2024), measured across different tokenizers. Figure 5 reports the percentage of words with equal subword counts across domains for each diacritization pair. Figure 6 shows domain-wise Jaccard similarities for each diacritization pair. Figure 7 shows cosine similarities of word-level representations from the first layer, where we extract a single word vector by mean-pooling across its subword token representations.

B Computational Budgets

All open-weight model inference experiments were conducted using a single NVIDIA A100 GPU (80GB). The total compute time was at most 24 GPU hours. The largest model evaluated in our study has 13 billion parameters. For GPT-4o, we used OpenAI's Batch API, with total API usage costing no more than $15 USD.

C Prompts

For ArabicMMLU, we use the following prompt:

```
This is a {SUBJECT}. Select the
correct answer!
Question: {QUESTION}
{OPTIONS}
Answer:
```

Here, `SUBJECT`, `QUESTION`, and `OPTIONS` are dynamically filled for each example. The `OPTIONS` field consists of the answer choices, each prefixed by an English alphabet label (A, B, C, or D), followed by the corresponding option text.

For ArabCulture, we use the following prompt:

```
You are tasked with selecting
the most culturally appropriate
option based on the context
provided below.
Statement: {FIRST_STATEMENT}
Consider the cultural nuances and
choose the most suitable response
from the options provided.
Options:
{OPTIONS}
```

For AraTrust, we use the following prompt:

```
I will provide you with a question
and several options. Choose only
one answer.
Question: {QUESTION}
{OPTIONS}
Answer:
```

The `QUESTION` and `OPTIONS` fields are formatted as in ArabicMMLU.

D Inter-word Cosine Similarity

Figure 8 shows the layer-wise inter-word cosine similarities of word-level representations for `undiac-wild`, `undic-full`, and `wild-full` pairs. For each layer, we extract a single word-level vector by mean-pooling over its subword token representations. We then compute pairwise cosine similarities between representations of 3,000 words in the Wild2MaxDiacs dataset and report the average similarity as an aggregate measure.

E Diacritic Distribution Comparison

Figure 9 illustrates the similarity between the benchmark datasets and the PDD subsets (Elgamal et al., 2024) in terms of diacritic type distribution, quantified using Jensen-Shannon divergence (Lin, 1991).

F License

In Table 7, we list the license of the data and tools used in this work. All of them are used under their intended use.

Data/Tool	**License**
ArabicMMLU (Koto et al., 2024)	CC-BY-NC 4.0
ArabCulture (Sadallah et al., 2025)	CC-BY-NC-SA 4.0
AraTrust (Alghamdi et al., 2025)	MIT
LM Evaluation Harness (Gao et al., 2024)	MIT
CAMeL Tools (Obeid et al., 2020)	MIT

Table 7: License of the data and tools.

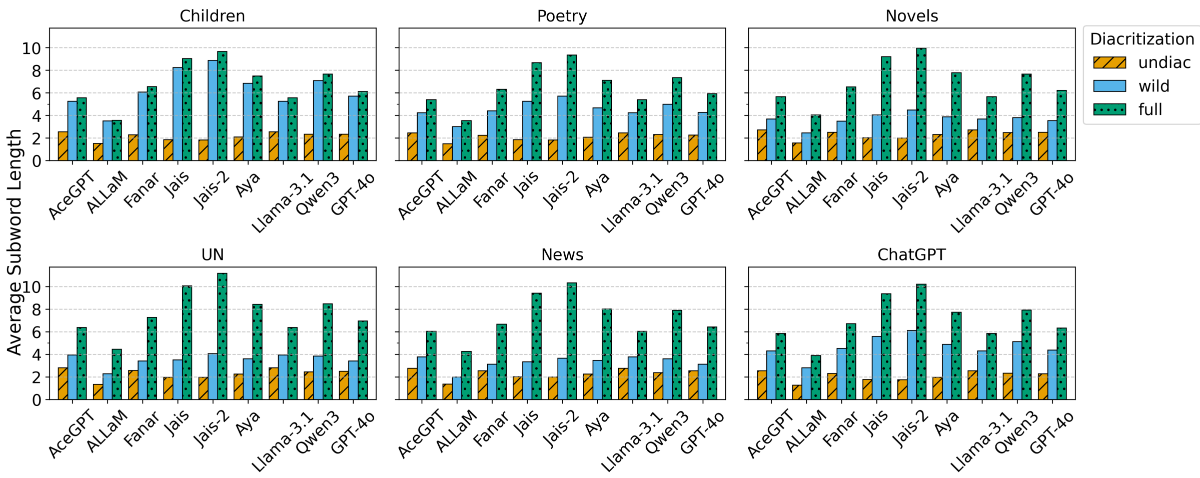

Figure 4: Average subword length for each domain in the Wild2MaxDiacs dataset (Elgamal et al., 2024), measured across different tokenizers.

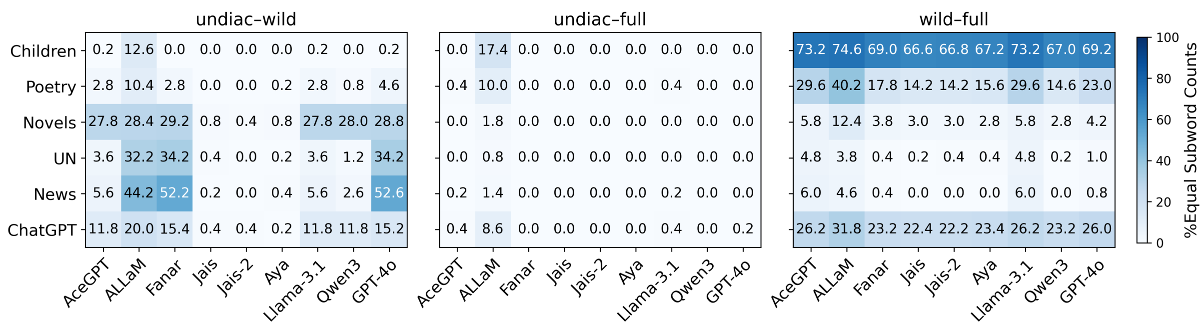

Figure 5: Percentage of words with equal subword counts across domains for each diacritization pair: `undiac-wild`, `undiac-full`, and `wild-full`.

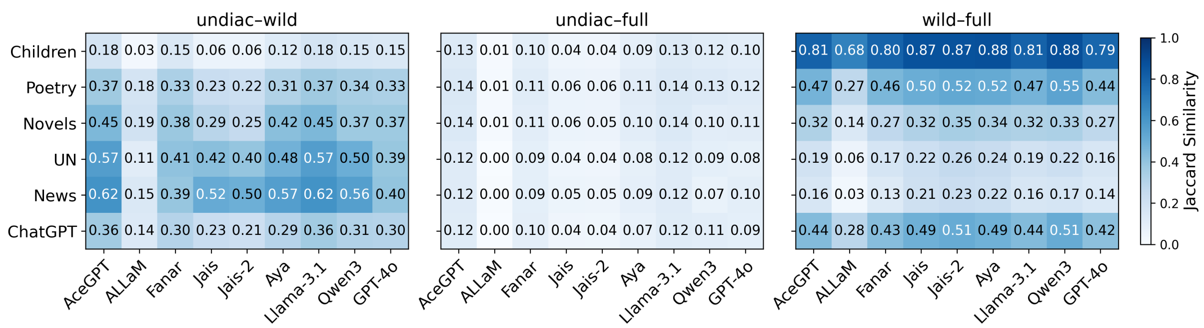

Figure 6: Domain-wise Jaccard similarities for `undiac-wild`, `undiac-full`, and `wild-full` parings.

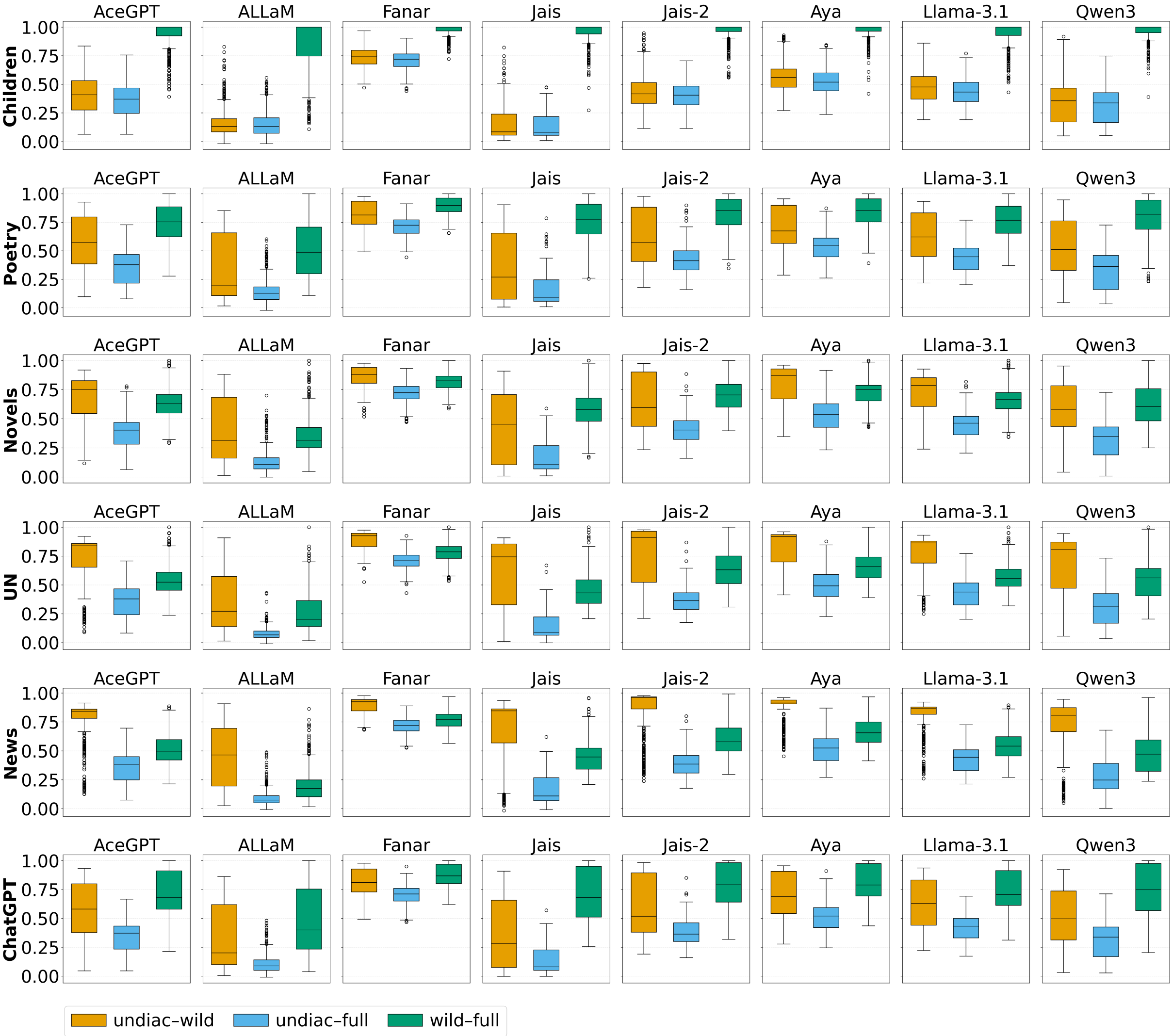

Figure 7: Cosine similarities of word-level representations from the first layer for `undiac-wild`, `undiac-full`, and `wild-full` parings.

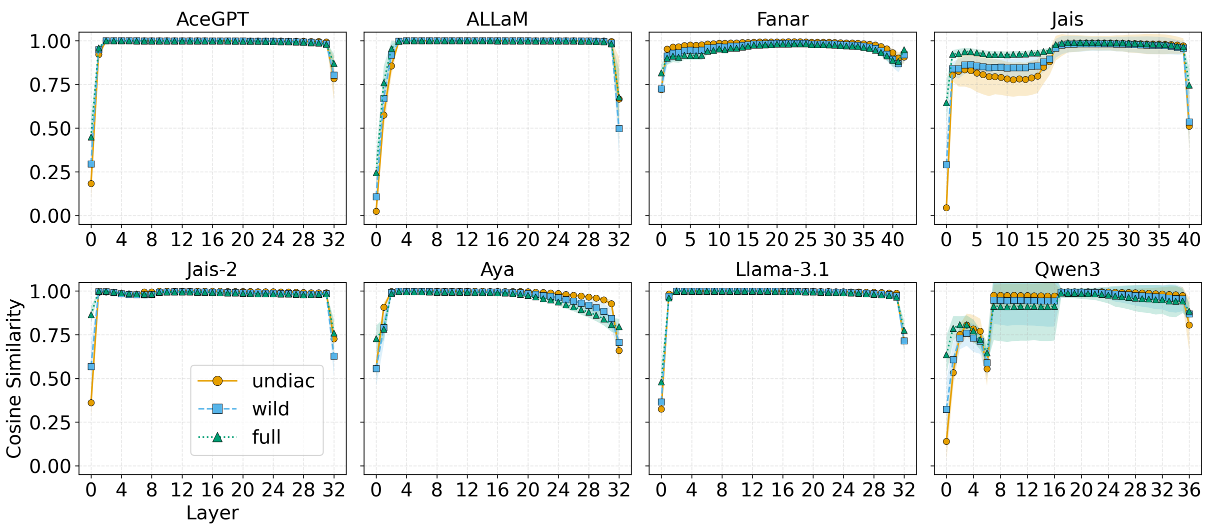

Figure 8: Layer-wise inter-word cosine similarities of word-level representations for `undiac-wild`, `undic-full`, and `wild-full` pairs. For each layer, we extract a single word vector by mean-pooling across its subword token representations. We then compute pair-wise cosine similarities among 3,000 words in the Wild2MaxDiacs dataset and aggregate these cosine similarities by taking the average.

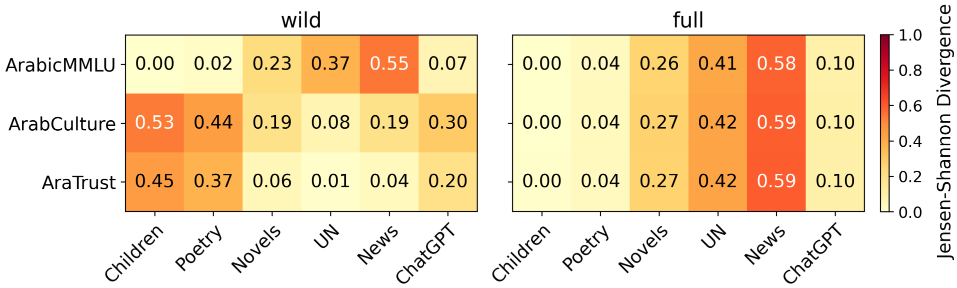

Figure 9: Jensen-Shannon divergence (JSD) of diacritic distributions between each benchmark dataset (rows) and the PDD dataset (Elgamal et al., 2024) (columns). Lower values indicate higher similarity in diacritic distributions. Each subfigure compares the divergence under the `wild` (left) and `full` (right) diacritization settings.