Partial Diacritization: A Context-Contrastive Inference Approach

Anonymous ACL submission

Abstract

Diacritization plays a pivotal role in improving readability and disambiguating the meaning of Arabic texts. Efforts have so far focused on marking every eligible character (Full Diacritization). Comparatively overlooked, Partial Diacritzation (PD) is the selection of a subset of characters to be annotated to aid comprehension where needed. Research has indicated that excessive diacritic marks can hinder skilled readers—reducing reading speed and accuracy. We conduct a behavioral experiment and show that partially marked text is often easier to read than fully marked text, and sometimes easier than plain text. In this light, we introduce Context-Contrastive Partial Diacritization (CCPD)—a novel approach to PD which integrates seamlessly with existing Arabic diacritization systems. CCPD processes each word twice, once with context and once without, and diacritizes only the characters with disparities between the two inferences. Further, we introduce novel indicators for measuring partial diacritization quality to help establish this as a machine learning task. Lastly, we introduce `TD2`, a Transformer-variant of an established model which offers a markedly different performance profile on our proposed indicators compared to all other known systems.[1]

1 Introduction

The Arabic language is central to the linguistic landscape of over 422 million speakers. It plays a pivotal role in the religious life of over a billion Muslims (Mijlad and El Younoussi, 2022). As in other impure **abjad** writing systems, the Arabic script omits from writing some phonological features, like short vowels and consonant lengthening. This can affect reading efficiency and comprehension. Readers use context from neighbouring words, the domain topic, and experience with the language structure to guess the correct pronunciation and disambiguate the meaning of the text.

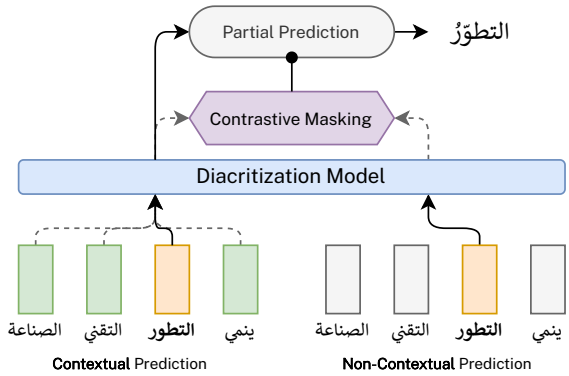


Figure 1: The proposed system employs a contextual diacritization model in two modes. *(Left)* Model receives a word with its surrounding context, *(Right)* Model receives the word in isolation. *(Top)* The outputs are contrasted to select a subset of the text to diacritize.

The Arabic NLP community has noticeably focused on the task of Full Diacritization (**FD**)—the modeling of diacritic marks on every eligible character in a text (for example: Darwish et al., 2017; Mubarak et al., 2019; AlKhamissi et al., 2020). This is especially useful in domains where ambiguities are not allowed, or where deducing the correct forms might pose challenges for non-experts. Such domains may include religious texts, literary works like poetry, or educational material.

There are benefits for human readers, like facilitating learning. However, prior research suggests that extensive diacritization can inadvertently impede skilled reading by increasing the required processing time (Taha, 2016; Ibrahim, 2013; Abu-Leil et al., 2014; Midhwah and Alhawary, 2020; Roman and Pavard, 1987; Hermena et al., 2015). Nevertheless, diacritics are important morphological markers, even when excessive, and may benefit automated systems in language modeling, machine translation (MT), part-of-speech tagging, morpho-

[1] A demo of this work will be available upon publication.

	System	Sentence
		Full Diacritization
	Truth	سَوْفَ نَحْيَا هُنَا .. سَوْفَ يَحْلُو النَّغَمُ
		Partial Diacritization
–MV, hard–	TD2	سوفَ نحيا هُنا .. سوفَ يحلو النغَمْ
	D2	سوف نحيَا هنا .. سوفَ يحلو النُّغْمَ
—D2–SP—	hard	سوف نحيا هنا .. سوفَ يحلو النغم
	soft > 0.1	سوف نحيَا هنا .. سَوف يحلو النَّغم
	soft > 0.01	سَوف نحيَا هنا .. سَوْفَ يحلو النَّغم

Table 1: System Outputs on text from the Tashkeela test-set. "soft" methods have an adjustable threshold. **MV** uses majority-voting, while **SP** (Single-Pass) doesn't (see Appendix A.2 and Section 6.2). The line translates into: "We will live here. The singing will be sweet."

Glyph	Name	Type	BW	IPA
ُ	dammah	ḥarakāt	*u*	/hu/
َ	fathah	ḥarakāt	*a*	/ha/
ِ	kasrah	ḥarakāt	*i*	/hi/
ْ	sukūn	sukūn	*o*	/h./
ٌ	dammatain	tanwīn	*N*	/hun/
ً	fathatain	tanwīn	*F*	/han/
ٍ	kasratain	tanwīn	*K*	/hin/
ّ	shaddah	shaddah	∼	/h:/

Table 2: Primary Arabic Diacritics on the Arabic letter *h*. **BW** is the Buckwalter transliteration of the vowel or syllable. Adapted from AlKhamissi et al. (2020).

logical analysis, acoustic modeling for speech recognition, and text-to-speech synthesis. For an example in MT, see Fadel et al. (2019); Habash et al. (2016); Alqahtani et al. (2016); Diab et al. (2007).

1.1 Contributions

Context-Contrastive Partial Diacritization We propose a novel method for Partial Diacritization (**PD**) which seamlessly utilizes existing Arabic FD systems. We exploit a statistical property of Arabic words wherein readers can guess the correct reading of most unmarked words with *minimal context*. To select the letters to mark, CCPD sees each word twice: (1) within its sentence context, and (2) as an isolated input with no context. The two predictions are combined to retain only those diacritics which present comparatively new information that may aid reading comprehension (Section 4).

Human Evaluation We conduct a behavioral experiment to compare ease of reading for text with all, some, or no diacritics. Diacritics are selected via a neural model mask to simulate native partial marking. The results support prior work on the effect of different degrees of diacritcs on reading efficiency and comphrension, and is the motivation for this work (see Section 8).

Performance Indicators We introduce a set of automatic indicators to gauge the performance and usefulness of our method on several public models, in light of the scarcity of supervised labeled partially diacritized datasets (see Section 5).

Transformer D2 And last, we present the **TD2** model, a Transformer variant of D2 (AlKhamissi et al., 2020), which shows markedly improved Partial DER performance at 5.5 % compared to 11.2 %, even while marking a *larger* percentage of text at 24.6 % compared to 6.5 % for D2 (see Section 6.1).

2 Motivation

Previous research underscores the substantial influence of diacritization on the reading process of Arabic text. Skilled readers exhibit a tendency to read highly diacritized text at a slower pace compared to undiacritized. The effect on reading speed and accuracy has been substantiated across numerous studies involving diverse age groups and linguistic backgrounds (Taha, 2016; Ibrahim, 2013; Abu-Leil et al., 2014; Midhwah and Alhawary, 2020; Roman and Pavard, 1987; Hermena et al., 2015; Hallberg, 2022). In addition, studies monitoring eye movements during reading link the increased reading times for diacritized text to an increase in fixation frequency and duration—a key indicator of the word identification process (Roman and Pavard, 1987; Hermena et al., 2015). *Partial Diacritization* may thus lead to a better reading experience for a wide range of readers, including in cases with dyslexia or visual impairments.

3 Arabic Preliminaries

Diacritics in Modern Arabic Orthography The inventory of diacritics used in modern typeset Arabic comprises at least four functional groups.

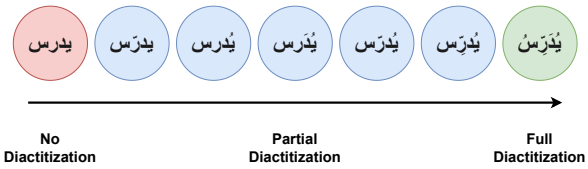


Figure 2: Seven possible ways among others of writing the word `yudarrisu` (English: He teaches) with varying levels of diacritc coverage.

First are ḥarakāt and sukūn, which indicate vowel phonemes or their absence. Then tanwīn and shaddah indicate case-inflection morphemes and consonant lengthening. There are other marks, e.g. for recitation features in the Qur'an. Only the marks in **Table 2**, part of modern Arabic orthography, are discussed here. Ḥarakāt are vowel diacritics that provide vowel information. Sukūn indicates the absence of a vowel, which indicates consonant clusters or diphtongs. Tanwīn diacritics indicate the phonemic pair of a vowel and the consonant /n/ at the end of words, serving case inflection purposes. These diacritics play essential roles in Arabic orthography and pronunciation (Hallberg, 2022).

Modes of Diacritization. Since diacritics are optional and their usage can vary widely—practical patterns have emerged. Hallberg (2022) identifies seven modes of diacritization. These modes can be ordered based on the quantity of diacritics used, ranging from no diacritization to complete diacritization. Deeper levels, like *complete* diacritization, are less frequently used than shallower levels (no diacritization, and so on). Examples in Figure 2.

4 Methods

4.1 Full Diacritization Models

We utilize deep neural sequence models for character diacritization. We test the pretrained models `D2`, `Shakkelha` and `Shakkala` (AlKhamissi et al., 2020; Fadel et al., 2019; Barqawi, 2017). Further, we introduce the `TD2` model: a Transformer variant of the LSTM-based `D2`. A description of `TD2` and the `D2` architecture can be found at Section 6.1.

4.2 What Context Do Models See?

Context extraction. Let $\mathsf{ctxt}(x; \mathbb{X})$ contextualize x from $\mathbb{X}$. Then the word and letter context extractors are functions which use the surrounding words and letters around positions i and j.

Words around position i

$$[w_i]_T = \mathsf{ctxt}(w_i; \{w_t \in s_{i,T}\}) \quad (1)$$

$$[\ell_j \mid w_i] = \mathsf{ctxt}(\ell_j; \{\ell_t \in w_i\}) \quad (2)$$

Letters in Word i

Where $s_{i,T}$ contains all the words in a *segment* $s_{i,T} = \{w_t \mid t \in i \pm T\}$ as used in training `D2`/`TD2`.

Contextual Prediction. Let f use W and L for word and letter contexts as follows:

$$f(\mathsf{W} = [w_i]_T, \mathsf{L} = [\ell_j \mid w_i]) \quad (3)$$

Where $[x_n]_T$ refers to the context around n.

Single Word Prediction sees only the current word at position i. Notice the w_i instead of $[w_i]_T$.

$$f(\mathsf{W} = w_i, \mathsf{L} = [\ell_j \mid w_i]) \quad (4)$$

We use "word" to refer to syntactically separated

Thus the two modes are defined as follows: f_1 is single-word application, while f_c uses sentence context, given the same model f.

$$f_c(i,j) = f(\mathsf{W} = [w_i]_T, \mathsf{L} = [\ell_j \mid w_i]) \quad (3)$$

$$f_1(i,j) = f(\mathsf{W} = w_i, \mathsf{L} = [\ell_j \mid w_i]) \quad (4)$$

4.3 *Context-Contrastive Partial Diacritization*

The function f_1 takes each word in isolation. If its prediction is correct, we assume that this reading is common and easy to guess by a human reader. By contrast, f_c utilizes word context *within a sentence*. If a word has multiple possible readings which the sentence disambiguates, we expect f_c with sentence context to out-perform f_1.

Following from these premises, we propose the following method to combine both sources of information to diacritize text partially, with justification.

CCPD: Algorithm Using f_c and f_1 probabilities, for a word w_i and character ℓ_j in dataset $\mathcal{D}$, the system outputs a diacritic according to CCPD in:

$$\mathsf{CCPD}(i,j) := \begin{cases} \arg\max_{\mathsf{class}} f_c(i,j) & \leftarrow i,j \in \mathcal{D}_S, \\ \varnothing & \leftarrow \text{otherwise} \end{cases} \quad (5)$$

$$\begin{aligned} \mathcal{D}_S &:= \{i,j \in \mathcal{D} \mid \mathsf{mark}(i,j)\} && \text{selected} \\ \mathcal{D}_U &:= \mathcal{D} \setminus \mathcal{D}_S && \text{unmarked} \end{aligned} \quad (6)$$

$$\mathsf{mark}_{\text{soft}}(i,j) := \left[\max_{\text{class}} f_c - \max_{\text{class}} f_1 > \theta\right](i,j) \quad (7)$$

$$\mathsf{mark}_{\text{hard}}(i,j) := \left[\arg\max_{\text{class}} f_c \neq \arg\max_{\text{class}} f_1\right](i,j) \quad (8)$$

In hard mode, the characters where applications with and without context *agree* are not assigned diacritics (regardless of the correctness of the prediction; unknown during inference). Otherwise, the prediction of the contextual f_c model is returned.

In soft mode, only the characters where the logit of f_c's prediction exceeds the logit of f_1's by a margin θ receive a diacritical mark.

5 Performance Indicators

To measure performance on the partial diacritization task, we propose four indicators as approximate gauges for model alignment with human expectation. These indicators aim to address the challenge of the limited availability of high-quality partially diacritized test datasets, providing valuable insights into the model's capabilities.

5.1 SR: Selection Rate

The proportion of characters assigned a diacritic by Function (5). Literature shows partial diacritization hovers around 1.2 %–9.5 % coverage in some professionally published books (Hallberg, 2022), while other research suggests that deliberate partial diacritization by native speakers results in a rate around 19 %–26 % (Esmail et al., 2022). **SR** is:

$$\mathsf{SR}(\mathcal{D}) = \frac{1}{|\mathcal{D}|} \sum_{i,j \in \mathcal{D}} \mathbb{1}\big(\overbrace{\mathsf{CCPD}(i,j)}^{\text{Final (selective) system output}} \neq \varnothing\big) \tag{9}$$

Full corpus; Word i, char j

$\mathbb{1}(x \neq 0)$ are Logical expressions: 1 if true, else 0.

5.2 Scoped Diacritic Error Rates $\mathsf{DER}(f, \mathcal{D})$

DER in traditional literature is the ratio of diacritics erroneously predicted over all eligible letters in a corpus. Trivially, we parameterize DER by the system f and corpus (subset) $\mathcal{D}$ such that:

$$\mathsf{DER}(f, \mathcal{D}) = \frac{1}{|\mathcal{D}|} \sum_{i,j \in \mathcal{D}} \mathbb{1}(\overbrace{f}^{\text{Predictor}} \neq \overbrace{\mathsf{gt}}^{\text{True labeler}})_{i,j} \tag{10}$$

(Subset of) letters "$\mathcal{D}$"; Word i, char j in $\mathcal{D}$

5.2.1 P-DER: Partial Diacritic Error Rate

We calculate the DER on $\mathcal{D}_S$: All characters in the corpus which were assigned diacritics by f_c, following Function (5). **P-DER** is:

$$\mathsf{PDER}(\mathcal{D}) = \mathsf{DER}(\overbrace{f_c}^{\text{Contextual prediction}}, \overbrace{\mathcal{D}_S}^{\text{Marked letters}}) \tag{11}$$

5.2.2 B-DER: Basic Diacritic Error Rate

By the intuition in Section 4.3 regarding the non-contextual predictions of f_1, we calculate the DER on the whole corpus $\mathcal{D}$ and use **B-DER** as a proxy for human error on plain unmarked text.

$$\mathsf{BDER}(\mathcal{D}) = \mathsf{DER}(\overbrace{f_1}^{\text{Non-Contextual prediction}}, \overbrace{\mathcal{D}}^{\text{All letters}}) \tag{12}$$

5.3 R-DER: Reader Diacretics Error Rate

We combine the Partial DER and Hidden DER into one general measure of the total expected error when a human reads partially diacritized text. The Hidden DER indicator is proxy for the expected human error on undiacritized text, while Partial DER reports the actual model error on the diacritized text subset. We weigh both numbers by SR and report the Reading-DER (**R-DER**) as:

$$\begin{aligned}\mathsf{RDER}(\mathcal{D}) = (\mathsf{SR}) \quad & \mathsf{DER}(f_c, \mathcal{D}_S) \\ + (1 - \mathsf{SR}) \quad & \mathsf{DER}(f_1, \mathcal{D}_U)\end{aligned} \tag{13}$$

5.4 Redri: Reading-DER Improvement

Does the reading quality improve by adding a selected subset of diacritics from a system f? We report the difference of DER by f_c compared to f_1 over the marked subset $\mathcal{D}_S$. R-DER Improvement (**Redri**) indicates an improvement when positive, and a regression when negative. 'abs'

$$\mathsf{Redri}_{\text{abs}}(\mathcal{D}) = -\,\mathsf{RDER}(\mathcal{D}) + \mathsf{BDER}(\mathcal{D}) \tag{14}$$

$$\mathsf{Redri}_{\text{rel}}(\mathcal{D}) = 1 - \frac{\mathsf{RDER}(\mathcal{D})}{\mathsf{BDER}(\mathcal{D})} \tag{15}$$

6 Experimental Modeling

6.1 TD2: Transformer D2

The TD2 model is a Transformer adaptation of D2 (AlKhamissi et al., 2020). It comprises two encoders of 2 layers each, for tokens and characters. Both models use the same architecture. The feature and intermediate widths are (768, 2304). The token model is initialized from the pretrained weights of layers 1-2 of *CAMeL-Lab/bert-base-arabic-camelbert-mix-ner* (Inoue et al., 2021) as provided by HuggingFace[2], while the character model is initialized from layers 3-4.

[2] HuggingFace: Wolf et al. (2020)

	System	SR	Redri (rel)	P-DER	R-DER	B-DER	DER	WER
	REFERENCE RANGE	$13 \pm 8\,\%$	$\uparrow \pm 100\%$	$\downarrow 100\% \ldots 0\%$				
	Barqawi, 2017	8.6	6.1 (63)	17.9	†3.6	9.7	3.58	11.19
	Fadel et al., 2019 (big)*	8.9	**7.7 (83)**	7.8	†**1.6**	9.3	**1.60**	**5.08**
	AlKhamissi et al., 2020 - **D2**	6.5	*5.3 (75)*	11.2	†*1.8*	7.1	*1.85*	*5.53*
—Ours—	↪ (SP, hard)	6.8	**5.1 (72)**	15.0	†**2.0**	7.1	2.00	6.42
	↪ (SP, soft > 0.4)	5.9	2.8 (39)	**3.1**	4.3			
	↪ (SP, soft > 0.2)	**11.5**	4.3 (61)	*4.7*	2.8			
	↪ (SP, soft > 0.1)	**15.2**	4.8 (68)	5.1	2.3			
	↪ (SP, soft > 0.01)	21.8	**5.1 (72)**	4.9	**2.0**			
	TD2 (MV, hard)	*24.6*	**22.5 (90)**	*5.5*	†*2.4*	25.0	2.44	7.68

Table 3: Partial Diacritics Results on Tashkeela via the indicators outlined in Section 5. **SR** is Selection Rate, **P-DER** is Partial Diacritic Error Rate, **R-DER** is Reading-DER, while **Redri** is R-DER Improvement (eq. 14 & 15). A ratio of Redri to B-DER is also reported in parentheses (higher is better). We report DER and WER as well. **(SP)** (SinglePass) results run inference on the whole sentence at once, with no segmentation or majority voting. Fadel et al. (2019)* uses the `lstm-big-20` configuration and uses extra training data beyond the Tashkeela `train` split; it is part of the public code release, but is not reported on in the original paper. † Under hard mode letter selection, R-DER is equal to $\mathrm{DER}(f_c, \cdot)$, which follows from eq. (13).

6.2 Single-Pass Inference

For recurrent (non-Transformer) models like D2, we also test inference without segmentation and majority voting wherein each sentence is passed once and in full to the model (SP). This can save inference time (by avoiding overlapping windows), but sacrifices some reliability (as the model is trained on smaller window sizes).

6.3 Other Models

Shakkala (Barqawi, 2017) and Shakkelha (Fadel et al., 2019) are LSTM-based models that view sentences as a sequence of characters. In contrast, D2 and TD2 view sentences as word sequences.

7 Results

7.1 Full Diacritization Performance

Following prior work, we report the Diacritic Error Rate (DER) and Word Error Rate (WER) twice in Table 3, once including case-endings (CE) and once excluding them. CE denotes the diacritics located at the word's end, usually determined by the word's syntactic role. Predicting these diacritics is more challenging compared to core-word diacritics (CW), which specify lexical choices and are found elsewhere within the word.

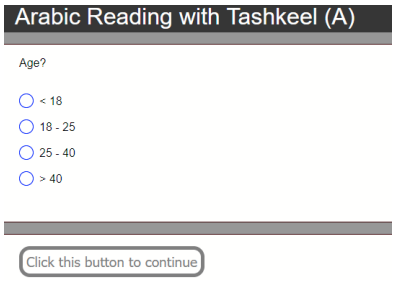
7.2 Partial Diacritization Performance

We notice that the recurrent D2 is conservative in its selection with **SR** at 6.46% versus TD2's 24.61%. TD2 performs terribly in non-contextual mode with **B-DER** at 25%. It definitively recovers however, with a **Redri** of 22.5%—meaning that only 2.4% of all letters remain "expectedly erroneous". Both systems agree that much of the text (at least 75%) is reasonably guessable by the reader without annotation. This is tricky to infer. Notice that both D2 and TD2 correct most of their B-DER error via CCPD, implying that most erroneous letters were accurately selected. We may claim that the expected B-DER between 75% and 93% (the extremes of the two systems). **SR** can be tweaked in soft selection mode and generally lies within $15 \pm 10\%$ or around the rates reported in Section 5.1.

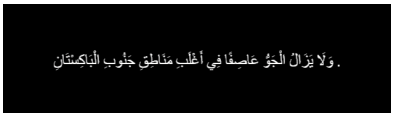
Assuming that the selected letters are the hardest to guess for a reader, this suggests that D2-SP models can disambiguate the text by 39.4% to 71.8% ($\mathrm{Redri}_{\mathrm{rel}}$: relative improvement in DER) by marking only 5.9% to 21.8% of it. While D2 and TD2 show a 71.8% and 90% improvement at 6.8% and 24.6% in *hard* mode, respectively. Despite a higher selection rate, TD2 manages to score better in P-DER than D2.

8 Behavioral Experiment on Partial Diacritization

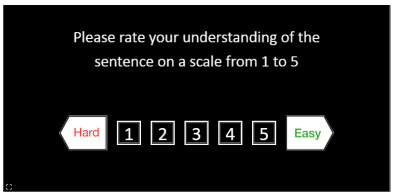
We start with the hypothesis that reading partially diacritized text may be easier than reading fully diacritized text, while providing some benefit in disambiguation or ease-of-reading.



(a) Demographic questions, like Age and Arabic Proficiency.



(b) Examples are shown for 2-5 seconds with zero/some/all diacritics, in white typeface on black background.



(c) Participants have to socre their understanding of the previous sentene between 1 and 5.

Figure 3: Screenshots of Behavioral Experiment Steps. (3a) Demographic info is collected on: Age, Number of Languages Spoken, Arabic Proficiency, and Native Arabic Dialect Spoken. (3b) Each participant was assigned one of 3 different test versions. All versions include the same sentences, but differ in which sentences are assigned how many diacritics. (3c) Participants rate their understanding of an example from 1 to 5. Participants were informed that the anonymized scores and metadata will be collected and used in an academic work.

To test this hypothesis, we conduct a behavioral experiment using machine-predicted partial diacritization *masks*. The diacritical marks themselves are the true labels. This is to ensure that the presented text is accurate, even if the model output is inaccurate. The question is then: *Does CCPD between f_c and f_1 work well for choosing the letters to be diacritized?* See the results in **Figure 4**.

8.1 Behavioral Experiment Setup

Demographics We utilize the PsyToolkit online platform[3] to conduct a behavioral experiment aimed at assessing the impact of different levels of diacritization (modes) on reading speed and accuracy. Data was gathered from a group of 15 participants, covering various age groups and native speakers of different Arabic dialects. The majority of participants (10) fell within the 25–40 age group; 2 were below 25, and 3 were above 40. All participants spoke at least 2 languages, while 6 spoke 3 or more. Eight participants spoke either Gulf (4) or Maghrebi (4) Arabic natively, and 7 spoke Egyptian (2), Levantine (2), or Sudanese (2). One did not report a native dialect. All participants reported being native speakers of Arabic.

Dataset To assess the impact of diacritics, we select 30 sentences from various domains in the WikiNews testset (Darwish et al., 2017). Each sentence is presented in three variants with Zero/Partial/Full diacritics. The Partial variant is produced using our CCPD algorithm and the `D2-SP` model in hard selection mode to mask out easy or guessable ground truth diacritics.

Data Splits The data is split into 3 buckets of 10 sentences each. Each bucket has 3 versions as outlined previously. We create 3 test sets such that the buckets are rotated and each appears exactly once in any mode (Zero/Partial/Full). For example, the first bucket would appear in the test sets A, B, C in its Full, Zero, and Partial variants. This is done to ensure that any participant sees a sentence exactly once, to avoid biasing their rating by repeated exposure. This ensures also that each sentence is seen equal times in each mode.

Timing and Scoring Participants are shown each sentence for a few seconds proportional to the word count of the sentence: $1/4\ |\text{words}|$. Then the participants are prompted to rate their comprehension by a score from 1 to 5, where 1 indicates difficulty and 5 indicates ease in understanding the sentence. The design intention is to measure the reading experience of a native speaker when scanning a non-technical text in Modern Standard Arabic.

8.2 Findings

Figure 4 illustrates the aggregated self-reported reading comprehension scores, normalized by the score of the no diacritization (Zero) mode, for each diacritization mode. The results indicate that Full diacritization hinders reading accuracy when participants are provided with a limited amount of time to read the text, aligning with findings discussed in Section 2. In contrast, the Partial diacritization mode sometimes enhances reading comprehension performance compared to the Zero mode.

The data from this behavioral experiment will be available on the paper's GitHub repository[4]

[3] PsyToolkit: Stoet (2010, 2017)

[4] The code and data will be available upon publication

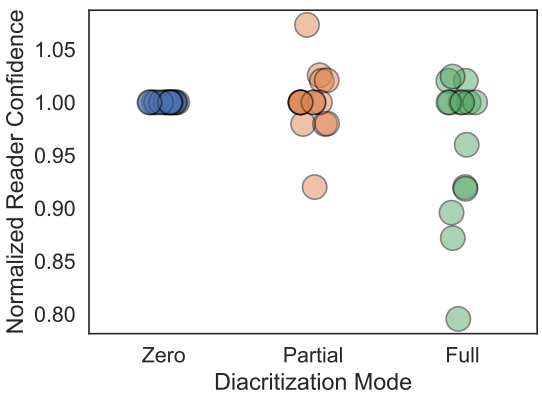


Figure 4: **Behavioral Experiment**: Self-reported scores for reading comprehension of sentences with Zero/Partial/Full diacritics. Scores are aggregated per mode and normalized by the score of the Zero mode of each participant, making the results comparable across participants. Notice the higher score points for partially diacritized samples (with some regressions), compared to the big regression for fully diacritized text.

9 Related Work

The literature has dealth with both types of diacritization: Full (FD) and Partial (PD). The majority has focused on FD. In PD, a distinct approach that seeks to augment reading comprehension by incorporating only the minimal requisite diacritics, we note a few works focus (Almanea, 2021; Mijlad and El Younoussi, 2022). Other works take a morphological analysis approach (Obeid et al., 2022, 2020; Alqahtani et al., 2016; Shahrour et al., 2015; Habash and Rambow, 2007).

9.1 Full Diacritization

Non-Neural Methods This class of methods combines linguistic rules with non-neural modelling techniques, such as Hidden Markov Models (HMMs) or Support Vector Machines (SVMs), which were widely employed over a decade ago. For example, Elshafei et al. (2006) utilize an HMM to predict diacritization based on bigram and trigram distributions. Bebah et al. (2014) extract morphological information and utilize HMM modeling for vowelization, considering word frequency distribution. (Shaalan et al., 2009) combine lexicon retrieval, bigram modeling, and SVM for POS tagging, addressing inflectional characteristics in Arabic text. Darwish et al. (2017) operate diacritization in two phases, inferring internal vowels using bigrams and handling case endings via an SVM ranking model and heuristics. Said et al. (2013) follow a sequence-based approach, involving autocorrection, tokenization, morphological feature extraction, HMM-based POS tagging, and statistical modeling for handling OOV terms. Zitouni and Sarikaya (2009) approaches the problem using maximum entropy models.

Neural Methods Recent literature has focused more on neural-based systems. Belinkov and Glass (2015) were the first to show that recurrent neural models are suitable candidates to learn the task entirely from data without resorting to manually engineered features such as morphological analyzers and POS taggers. AlKhamissi et al. (2020) used a hierarchical, BiLSTM-based model that operates on words and characters separately with a cross-level attention connecting the two—enabling SOTA task performance and faster training and inference compared to traditional models. Several other papers as well used recurrent-based models for the FD task (Al-Thubaity et al., 2020; Darwish et al., 2020; Gheith Abandah, 2020; Fadel et al., 2019; Boudchiche et al., 2017; Moumen et al., 2018), others used the Transformer architecture (Mubarak et al., 2019), while others used ConvNet architectures (Alqahtani et al., 2019b).

Hybrid Methods Some works have combined neural and rule-based or other methods to improve Arabic diacritization (Alqahtani et al., 2020; Abbad and Xiong, 2020; Darwish et al., 2020; Alqudah et al., 2017; Hifny, 2018).

9.2 Partial Diacritization

Similar to this work, previous research has explored the selective diacritization of Arabic text. Alnefaie and Azmi (2017) harnessed the output of the MADAMIRA morphological analyzer (Pasha et al., 2014) and leveraged WordNet to generate word candidates for diacritics. This work focused on resolving word ambiguity through statistical and contextual similarity approaches to enhance diacritization effectiveness. (Alqahtani et al., 2019a) focused on selective homograph disambiguation, proposing methods to automatically identify and mark a subset of words for diacritic restoration. Evaluation of various strategies for ambiguous word selection revealed promising results in downstream applications, such as neural machine translation, part-of-speech tagging, and semantic textual similarity, demonstrating that partial diacritization effectively strikes a balance between homograph disambiguation and mitigating sparsity effects. Esmail

et al. (2022) employ two neural networks are—one considering the entire sentence and the other considering the text read so far—to predict diacritics. Partial diacritization decisions are made based on disagreements between the two networks, favoring the holistic sentence-based approach.

10 Discussion

In this work, we have presented a novel approach for partial diacritization using context-contrastive prediction. The motivation behind using partial diacritization stems from a large body of literature on the impact of diacritization coverage on the reading process of Arabic text, and the lack of research done in this area.

Context-Contrastive Prediction By considering contextual information alongside non-contextual predictions, our approach builds on the role of diacritics in text, particularly in the context of the Arabic script as used by humans where accuracy is not the only consideration. The result is a simple, efficient, and configurable method which can be integrated with any existing diacritization system. It opens doors to enhanced diacritization accuracy and selection. The indicators we introduce provide valuable signals for evaluating such advancements.

The Effect of Diacritics on Reading A significant motivation for our work is grounded in prior research that underscores the substantial influence of diacritization on the reading process of Arabic text. Multiple studies have consistently shown that skilled readers tend to read extensively diacritized text at a slower pace compared to undiacritized text. This phenomenon, supported by research involving diverse age groups and linguistic backgrounds, has significant implications for reading accuracy and comprehension (Taha, 2016; Ibrahim, 2013; Abu-Leil et al., 2014; Midhwah and Alhawary, 2020; Roman and Pavard, 1987; Hermena et al., 2015; Hallberg, 2022). Other studies also suggest that diacritization not only affects reading speed but also has a direct impact on reading accuracy (Roman and Pavard, 1987; Hermena et al., 2015). In this work, we complement this large body of research by conducting a behavioral experiment that employed our CCPD approach to partially diacritize news headlines from the WikiNews test set. Our results further show that partially diacritized text not only is easier to read than fully diacritized text, but in some cases can lead to better understanding than no diacritization at all.

Implications An exciting potential for this work is optimizing the reading experience for different groups of readers with different needed levels of diacritization, including people with dyslexia and visual impairments. Since diacritization plays an important role in disambiguating the meaning of the text, it is crucial to intelligently select the ones that aid readability and comprehension of Arabic text without excessive marking; making it accessible and accommodating to a broader audience. This aligns with the broader goals of inclusivity and accessibility in NLP applications.

Future Directions In this work, we proposed a battery of indicators to gauge the performance of partial diacritization systems. However, further research is needed to gather high-quality partial diacritization benchmarks, allowing the incorporation of additional metrics or indicators for assessing model performance. Evaluating our system's performance on other languages that employ diacritical marks in their core orthography would clarify the general applicability of our approach.

11 Conclusion

In conclusion, our CCPD approach for partial diacritization using context-contrastive prediction not only contributes to the field of diacritization but also has far-reaching implications for optimizing the reading experience of Arabic text. Grounded in previous research on the influence of diacritization on reading—further supported by behavioral experiments conducted in this work—our approach paves the way for advancements that benefit readers of diverse backgrounds and abilities. Since our approach integrates seamlessly with existing Arabic diacritization systems, it can be used by used by models trained on domain-specific text as well. We also propose a battery of performance indicators to measure the competence of partial diacritization systems due to the lack of publicly available partial diacritization benchmarks. Finally, we introduce a new diacritization model, `TD2`—a Transformer adaptation of the `D2` model which offers a different performance profile as measured by our proposed indicators. As Arabic NLP continues to evolve, our approach serves as a promising direction for enhancing diacritization and its impact on text accessibility and comprehension.

Limitations

The method presented in this work is tested on a single human language. While we believe it should be able to generalize to other languages that need similar diacritics restoration, its utility needs further observation. In particular, it may be the case that partial diacritization patterns in languages besides Arabic are different in such a way that the premise of this work no longer holds (wherein ease of guessing is regarded as a major factor in omitting a diacritical mark).

In addition, this method is deliberately kept simple, which builds on an over-simplified view of the human task. Readers guess diacritics when reading, but they may do so while incorporating context via some simple or fast mechanisms. Whether that is at a deep level equivalent to guessing the reading of a single word in isolation remains to be seen. Our method uses a component, f_1, as a proxy for this process, which is likely relatively close in performance to the same mechanism in a human. Nevertheless, we make no claim that this is indeed the natural mechanism, nor that the proxy exhibits identical performance distribution to the natural mechanism.

References

Hamza Abbad and Shengwu Xiong. 2020. Multi-components system for automatic arabic diacritization. In *Advances in Information Retrieval*, pages 341–355, Cham. Springer International Publishing.

Aula Abu-Leil, David Share, and Raphiq Ibrahim. 2014. How does speed and accuracy in reading relate to reading comprehension in arabic? *Psicológica*, 35:251–276.

Abdulmohsen Al-Thubaity, Atheer Alkhalifa, Abdulrahman Almuhareb, and Waleed Alsanie. 2020. Arabic diacritization using bidirectional long short-term memory neural networks with conditional random fields. *IEEE Access*, 8:154984–154996.

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.

Manar M Almanea. 2021. Automatic methods and neural networks in arabic texts diacritization: a comprehensive survey. *IEEE Access*, 9:145012–145032.

Rehab Alnefaie and Aqil M. Azmi. 2017. Automatic minimal diacritization of arabic texts. *Procedia Computer Science*, 117:169–174. Arabic Computational Linguistics.

Sawsan Alqahtani, Hanan Aldarmaki, and Mona T. Diab. 2019a. Homograph disambiguation through selective diacritic restoration. In *WANLP@ACL 2019*.

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.

Sawsan Alqahtani, Ajay Mishra, and Mona Diab. 2019b. Efficient convolutional neural networks for diacritic restoration. *arXiv preprint arXiv:1912.06900*.

Sawsan Alqahtani, Ajay Mishra, and Mona Diab. 2020. A multitask learning approach for diacritic restoration. *arXiv preprint arXiv:2006.04016*.

Saba' Alqudah, Gheith A. Abandah, and Alaa Arabiyat. 2017. Investigating hybrid approaches for arabic text diacritization with recurrent neural networks. *2017 IEEE Jordan Conference on Applied Electrical Engineering and Computing Technologies (AEECT)*, pages 1–6.

Zaid Alyafeai, Maged S. Al-Shaibani, and Moataz Ahmad. 2023. Ashaar: Automatic analysis and generation of arabic poetry using deep learning approaches. *ArXiv*, abs/2307.06218.

Zerrouki Barqawi. 2017. Shakkala, Arabic text vocalization. `https://github.com/Barqawiz/Shakkala`.

Mohamed Ould Abdallahi Ould Bebah, Amine Chennoufi, Azzeddine Mazroui, and Abdelhak Lakhouaja. 2014. Hybrid approaches for automatic vowelization of arabic texts. *ArXiv*, abs/1410.2646.

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.

Mohamed Boudchiche, Azzeddine Mazroui, Mohamad Bebah, Abdelhak Lakhouaja, and Abderrahim Boudlal. 2017. Alkhalil morpho sys 2: A robust arabic morpho-syntactic analyzer. *Journal of King Saud University - Computer and Information Sciences*, 29:141–146.

Kareem Darwish, Ahmed Abdelali, Hamdy Mubarak, and Mohamed Eldesouki. 2020. Arabic diacritic recovery using a feature-rich bilstm model.

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.

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.

Moustafa Elshafei, Husni Al-Muhtaseb, and Mansour Alghamdi. 2006. Statistical methods for automatic diacritization of arabic text. *The Saudi 18th National Computer Conference. Riyadh*, 18:301–306.

Saeed Esmail, Kfir Bar, and Nachum Dershowitz. 2022. How much does lookahead matter for disambiguation? partial arabic diacritization case study. *Computational Linguistics*, 48:1–22.

Ali Fadel, Ibraheem Tuffaha, Bara' Al-Jawarneh, and Mahmoud Al-Ayyoub. 2019. Arabic text diacritization using deep neural networks. In *2019 2nd International Conference on Computer Applications Information Security (ICCAIS)*, pages 1–7.

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.

Asma Abdel-Karim Gheith Abandah. 2020. Accurate and fast recurrent neural network solution for the automatic diacritization of arabic text. *Jordanian Journal of Computers and Information Technology (JJCIT)*, 06(02):103 – 121.

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).

Andreas Hallberg. 2022. Variation in the use of diacritics in modern typeset standard arabic: A theoretical and descriptive framework. *Arabica*, 69(3):279 – 317.

Ehab Hermena, Denis Drieghe, Sam Hellmuth, and Simon Liversedge. 2015. Processing of arabic diacritical marks: Phonological-syntactic disambiguation of homographic verbs and visual crowding effects. *Journal of experimental psychology. Human perception and performance*, 41.

Yasser Hifny. 2018. Hybrid lstm/maxent networks for arabic syntactic diacritics restoration. *IEEE Signal Processing Letters*, 25(10):1515–1519.

Raphiq Ibrahim. 2013. Reading in arabic: New evidence for the role of vowel signs. *Creative Education*, 04:248–253.

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. *arXiv preprint arXiv:2103.06678*.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *International Conference on Learning Representations*.

Ali Al Midhwah and Mohammad T. Alhawary. 2020. Arabic diacritics and their role in facilitating reading speed, accuracy, and comprehension by english l2 learners of arabic. *The Modern Language Journal*, 104:418–438.

Ali Mijlad and Yacine El Younoussi. 2022. A comparative study of some automatic arabic text diacritization systems. *Advances in Human-Computer Interaction*, 2022.

Rajae Moumen, Raddouane Chiheb, Rdouan Faizi, and Abdellatif El Afia. 2018. Evaluation of gated recurrent unit in arabic diacritization. *International Journal of Advanced Computer Science and Applications*, 9.

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.

Ossama Obeid, Go Inoue, and Nizar Habash. 2022. Camelira: An Arabic multi-dialect morphological disambiguator. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 319–326, Abu Dhabi, UAE. 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.

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).

G. Roman and B. Pavard. 1987. A comparative study: How we read in arabic and french. In J.K. O'Regan and A. Levy-Schoen, editors, *Eye Movements from Physiology to Cognition*, pages 431–440. Elsevier, Amsterdam.

Ahmed Said, Mohamed El-Sharqwi, A. Fattah Chalabi, and Eslam Kamal. 2013. A hybrid approach for arabic diacritization. In *International Conference on Applications of Natural Language to Data Bases*.

Khaled Shaalan, Hitham M. Abo Bakr, and Ibrahim Ziedan. 2009. A hybrid approach for building Arabic diacritizer. In *Proceedings of the EACL 2009 Workshop on Computational Approaches to Semitic Languages*, pages 27–35, Athens, Greece. Association for Computational Linguistics.

Anas Shahrour, Salam Khalifa, and Nizar Habash. 2015. Improving Arabic diacritization through syntactic analysis. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1309–1315, Lisbon, Portugal. Association for Computational Linguistics.

Gijsbert Stoet. 2010. PsyToolkit: A software package for programming psychological experiments using Linux. *Behavior Research Methods*, 42(4):1096–1104.

Gijsbert Stoet. 2017. Psytoolkit: A novel web-based method for running online questionnaires and reaction-time experiments. *Teaching of Psychology*, 44(1):24–31.

Haitham Taha. 2016. Deep and shallow in arabic orthography: New evidence from reading performance of elementary school native arab readers. *Writing Systems Research*, 8(2):133–142.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. 2020. Huggingface's transformers: State-of-the-art natural language processing.

Taha Zerrouki and Amar Balla. 2017. Tashkeela: Novel corpus of arabic vocalized texts, data for auto-diacritization systems. *Data in Brief*, 11:147 – 151.

Imed Zitouni and Ruhi Sarikaya. 2009. Arabic diacritic restoration approach based on maximum entropy models. *Computer Speech & Language*, 23:257–276.

Appendix

A Experimenting with [Transformer] D2

A.1 Datasets

In this work, we train on data from the Tashkeela (Fadel et al., 2019) and Ashaar (Alyafeai et al., 2023) diacritization corpora. We test on the Tashkeela testset. The Tashkeela corpus has been collected mostly from Islamic classical books (Zerrouki and Balla, 2017) and contains mostly classical Arabic sentences. Ashaar is a corpus of Arabic poetry verses covering poems from different eras. **Table 4** shows the number of tokens for the Tashkeela and Ashaar[5] datasets.

	Train	Dev	Test
Tashkeela	$2,462,695$	$120,190$	$125,343$
Ashaar	$793,181$	$46,055$	$39,023$

Table 4: Token Counts for Each Data Split in the Tashkeela and Ashaar Diacritization Datasets

A.2 Majority Voting using a Sliding Window

Following prior work that utilizes an overlapping context window approach with a voting mechanism to enhance diacritic prediction for individual characters (Mubarak et al., 2019), we segment each input sentence into multiple overlapping windows. We present each segment to the model separately. This approach has proven effective, as localized context often contains enough information for accurate predictions. We use similar segmentation parameters to AlKhamissi et al. (2020)

Training For the training and validation sets we use a window of 10 words and a stride of 2.

Inference The same character may appear in different contexts, and therefore potentially result in different diacritized forms. We implement a popularity voting mechanism to narrow down the prediction. When a tie arises, we randomly select one of the top options. Testing is done with a sliding window of 20 words at a stride of 2.

A.3 Datapath of D2, TD2

The token model's output is averaged per word to result in exactly one feature vector (instead of one for each sub-word). This aggregated vector z^w is

[5] The reported numbers reflect dataset cleaning to keep only Arabic letters and diacritics.

concatenated with the character embedding of each character in the word and down-projected to the model feature size, to get character input $x^{c,w}$. The character encoder transforms the inputs $\{x^{c,w} \mid c \in \text{chars}(w)\}$ into character feature vectors $z^{c,w}$ which are passed to the final classifier.

Training The full model is tuned using the AdamW optimizer (Loshchilov and Hutter, 2019) with 0.2 dropout and 5×10^{-4} LR, and follows a linear schedule of 500 warmup steps and $10,000$ total training steps. The best checkpoint at $1,000$ step intervals is picked by the combined Tashkeela and Ashaar dev sets.

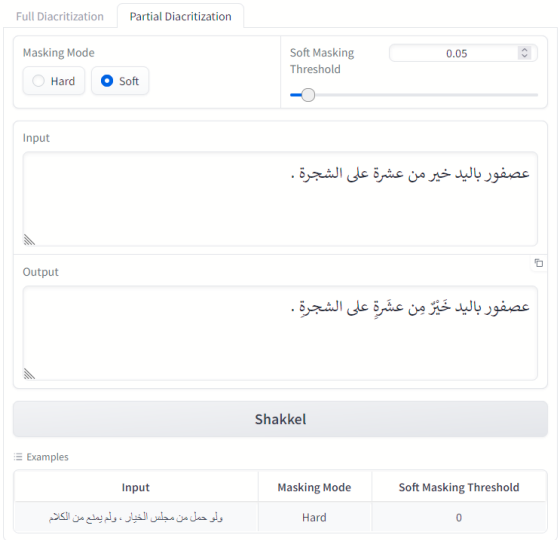
B Demo and Examples

We developed an online Demo on Huggingface[6] which allows choice between Full Diacritization, and Partial Diacritization with Hard and Soft modes. See Figures 5a and 5b. The demo supports the `D2/TD2` models (trained on Tashkeela and Tashkeela/Ashaar). Since they focus mostly on classical Arabic text, users are advised not to anticipate optimal performance when applying this model to Modern Standard Arabic (MSA) or informal Arabic dialects.

B.1 More Partial Diacritization Examples

The following examples are lines taken from a poem by Rami Mohamed (2016) called "We Will Stay Here". We applied the models as indicated in the System column. The input text included no diacritics at all. See Table 5.

(a) Full Diacritization UI: All predicted diacritics are returned. The model is capable of explicitly predicting no diacritics on a letter, but we did not notice this often.



(b) Partial Diacritization UI: Hard mode and Soft mode with threshold. This allows some rough control over the coverage percentage of output.

Figure 5: Demo UI hosted by HuggingFace supporting Full (Top) and Partial (Bottom) diacritization modes.

[6] huggingface.co/spaces

#	System	Text/Output	Error Count
	Truth	سَوْفَ نَبْقَى هُنَا .. كَيْ يَزُولَ الْأَلَمُ	
	Translation	We will Stay Here .. That the Pain may one day Cease	
1	TD2 (Full)	سَوْفَ نَبْقَى هُنَا .. كَيْ يَزُولَ الأَلَمُ	0
1	(MV, Hard)	سوفَ نبقى هُنا .. كيْ يزولَ الألمُ	0
3	D2 (Full)	سَوْفَ نَبْقَى هُنَا .. كَيْ يَزُولَ الْأَلَمُ	0
4	(MV, hard)	سوف نبقى هنا .. كي يزولَ الألمُ	0
5	(SP, soft > 0.05)	سَوف نبقى هنا .. كي يزولَ الألمُ	0
6	(SP, soft > 0.01)	سَوف نبقَى هنا .. كيْ يزولَ الألمُ	0
	Truth	فَلْنَقُمْ كُلُّنَا .. بِالدَّوَاءِ وَالْقَلَمِ	
	Translation	Let Us all Rise .. With Healing and Writing	
7	TD2 (Full)	فَلْنَقُمْ كُلُّنَا .. بِالدَّوَاءِ وَالْقَلَمِ	0
8	(MV, Hard)	فلْنَقُمْ كلُّنا .. بالدواء والقلمِ	0
9	D2 (Full)	فَلْنَقِمْ كُلّنَا .. بِالدَّوَاءِ وَالْقَلَمِ	1
10	(MV, hard)	فلنقمْ كلّنا .. بالدواء والقلمِ	0
11	(SP, soft > 0.01)	فلْنقمْ كلّنا .. بالدواء والقلمِ	0
	Truth	فَرْحَتِي وَصَرْخَتِي .. تَكَادُ تُسْمِعُ الْأَصَمَّ	
	Translation	My Joys and Screams .. Through Deafness Nearly Heard	
12	TD2 (Full)	فَرْحَتِي وَصَرْخَتِي .. تَكَادُ تَسْمَعُ الأَصَمَّ	2
13	(MV, Hard)	فرْحتي وصرختي .. تكادُ تسمع الأصمَّ	0
14	D2 (Full)	فَرْحَتِي وَصُرْخَتِي .. تَكَادُ تَسْمَعُ الْأَصَمَّ	3
15	(MV, hard)	فرحتي وصرختِي .. تكاد تَسمع الأصمَّ	0
16	(SP, soft > 0.01)	فَرحتِي وصرختِي .. تكَادُ تَسمعُ الأصم	1

Table 5: Examples use D2 from AlKhamissi et al. (2020). Notably, examples (6, 8, 11, 15) result in outputs similar to a native speaker's, aside from the rare error. For examples # 11 and 16, soft thresholds higher or lower than 0.01 did not change the output appreciably. The examples used are lines from the poem "We Will Stay Here" by Rami Mohamed (2016). All examples generated via the demo that will released upon publication.