ISSN 2712-0562
Sustainable Engineering and Innovation
Original Research
Vol. 6, No. 2, November 2024, pp.285-296
https://doi.org/10.37868/sei.v6i2.id305

Transformer-based automatic Arabic text diacritization

Ali Assad[1], Abdul Hadi M. Alaidi[2], Amjad Yousif Sahib[1], Haider TH. Salim ALRikabi[1], Ahmed Magdy[3]
[1]Electrical Engineering, College of Engineering, Wasit University, Wasit, Alkut, Iraq
[2]College of Computer Science and Information Technology, Wasit University, Wasit, Alkut, Iraq
[3]Electrical Engineering Department, Faculty of Engineering, Suez Canal University, Egypt

*Corresponding author E-mail: hdhiyab@uowasit.edu.iq

Received Oct. 13, 2023
Revised Jan. 29, 2024
Accepted Oct. 16, 2024

Abstract

In Arabic natural language processing (NLP), automatic text diacritization is a major obstacle, and progress has been slow when compared to other language processing tasks. Automatic diacritical marking of Arabic text is proposed in this work using the first transformer-based paradigm designed solely for this task. By taking advantage of the attention mechanism, our system is able to capture more of the innate patterns in Arabic, surpassing the performance of both rule-based alternatives and neural network techniques. The model trained with the Clean-50 dataset had a diacritic error rate (DER) of 2.03%, even though the model trained with the Clean-400 dataset had a DER of 1.37%. As compared to state-of-the-art results, the improvement for the Clean-50 dataset is minimal. However, for the larger Clean-400 dataset, it is a notable improvement, indicating that this approach can deliver more accurate solutions for applications requiring precise diacritical marks with larger datasets. Additionally, this method achieves a DER of 1.21% for the Clean-400 dataset, and it performs even better when given extended input text with overlapping windows.

Keywords: Automatic Arabic text diacritization, Transformer, Attention mechanism

1. Introduction

Natural language processing (NLP) tasks presented by Arabic exhibit unique obstacles due to the language's complicated orthographic system as well as its rich linguistic legacy. Along with the glottal stop (Hamza), the three long vowels (Alef, Waw, and Yeh) as well as twenty-five consonantal letters make up Arabic's complete alphabet. In addition, there are a total of 36 variants, with six variations for the letter Hamza, two variations for the letter Teh, as well as two variations for the letter Alef. Depending on where they appear in a word, each of these letters can take one of four possible forms.

To aid with both word pronunciation as well as comprehension, Arabic makes use of diacritics in addition to letters. A tiny mark that can be placed above or below the letter is called a diacritic. There are three primary categories of Arabic diacritics: vowel diacritics, nunation diacritics, as well as Shadda. Diacritical signs for vowels are Fatha, Damma, Kasra, as well as Sukun which denotes the lack of a vowel. Diacritics for nunation are written in a way that resembles the doubled forms of the short vowels Fathatan, Dammatan, as well as

Kasratan. A short vowel sound followed by an unwritten sound is represented by the Arabic letter "nunation" diacritics. Shadda is a diacritical mark that doubles consonants. A nunation or a single vowel diacritic may follow Shadda. With the letter "Dal" as an example, you can see the Arabic diacritics as well as how to pronounce them in Table 1.

Table 1. Primary Arabic diacritics (example is using the letter *Dal* (د) with the sound /d/)

Diacritic name	Example	Sound
Fatha	دَ	/da/
Damma	دُ	/du/
Kasra	دِ	/di/
Fathatan	دً	/dan/
Dammatan	دٌ	/dun/
Kasratan	دٍ	/din/
Sukun	دْ	/d/
Shadda	دّ	/dd/

The absence of diacritical marks (or Tashkeel) in numerous publications, such as magazines, books, as well as other materials, poses a substantial obstacle to Arabic text processing. It is widely believed—and rightfully so—that context may often clarify the intended meaning. This is left out to save time as well as effort. In order to clarify word meanings, guarantee correct pronunciation, as well as improve overall understanding, diacritical marks are necessary. But even native speakers could struggle to always employ the right diacritical marks in the right situations.

Texts with diacritical marks are required as training input for many NLP applications, including automated voice recognition, speech synthesis, automatic language translation, as well as many more [1]. To get around these problems, automatic Arabic text diacritization, which involves marking undiacritized Arabic text with diacritical marks, has grown into a crucial field of study in natural language processing. Historically, rule-based approaches to automatic Arabic text diacritization have been used, which frequently necessitate substantial language expertise as well as intricate rules that have been hand-crafted [2]. However, the complexity as well as context-dependence of diacritization are difficult for these methods to grasp.

Several artificial intelligence (AI) tasks, particularly those involving computer vision as well as natural language processing, have recently attained state-of-the-art outcomes thanks to deep learning techniques. Many natural language processing (NLP) applications, such as sentiment analysis, text summarization, as well as machine translation, have been transformed by the advent of deep learning approaches, especially transformer-based models. Because of its capacity to capture contextual linkages within sequences as well as long-range dependencies, the transformer model—introduced in [3]—has become extremely popular in natural language processing. A major shortcoming of the Arabic NLP community is the relatively low rate of adoption of deep learning methods. This constraint persists even in NLP specializations like sentiment analysis where deep learning techniques have become very popular.

Our objective in this study is to model a system for the automatic diacritization of Arabic text using the attention mechanism in a deep network. Transformers are able to accurately provide diacritized output by capturing the underlying patterns in Arabic text using the attention mechanism. As a data-driven method for automatic diacritization, transformer models are able to understand patterns of diacritization as well as generalize effectively to unseen material. More precise as well as efficient solutions are on the horizon thanks to the encouraging findings of using transformers for automatic Arabic text diacritization. The innovative sampling strategy that emphasizes training on underperforming samples as well as the deployment of a transformer-based model for Arabic text diacritization are our contributions.

2. Literature review

Various approaches have been taken to address the diacritization problem in Arabic text. Still, we need to keep working on this topic so we can get better at research. One can classify diacritization systems as either rule-based, statistical, or hybrid.

To address the diacritization issue, rule-based systems use a set of well-defined rules that are created by drawing on human knowledge found in dictionaries as well as morphological as well as grammatical patterns. While rule-based techniques are capable of producing adequate outcomes, their efficacy is contingent upon human knowledge as well as necessitates frequent rule revisions.

Shaalan delves into the use of a rule-based approach to build NLP systems, mostly with strong linguistic expertise, in [2]. He presents an Arabic-specific morphological analyzer as well as a syntax analyzer. In the absence of easily accessible large corpora, our work highlights the relevance of this technique, especially for languages having complicated morphology.

For an additional investigation in the same field, see [4]. Metwally et al. present a system for diacritizing Arabic text in this study. The system takes into account both syntactic as well as morphological diacritization of input word sequences. It doesn't matter where a word is in a phrase; its dictionary definition is the only determinant in morphological diacritization, which entails restoring diacritics based on that definition alone. Using context, native Arabic speakers usually have no trouble predicting these diacritics. The function of the word within the parsing tree of the sentence determines the syntactic diacritics. Even for native Arabic speakers, it can be difficult to accurately apply the intricate criteria that decide the syntactic diacritic for each word. There are a total of three levels that make up the system; Metwally et al.'s system, in its first layer, employs Hidden Markov models (HMM) to morphologically analyze previously encountered words. For terms that aren't in the lexicon, the morphological diacritization is handled by an external morphological analyzer at the next layer. For the syntactic diacritization of every word, the last layer uses conditional random fields (CRF). With a syntactic WER of 9.4% as well as a morphological WER of 4.3%, the suggested approach performs admirably. Typically used to evaluate state-of-the-art systems in this discipline, the evaluation made use of the widely known benchmark datasets from the LDC Arabic Treebank Part 3 project.

The creation of diacritics, an essential step in the phonetization process, is a connected issue. El-Imam [5] uses a dictionary of uncommon terms as well as a set of rules for the syntax as well as the morphology of the Arabic language to phonetize each letter in the text in an effort to use human expertise to solve this challenge. The percentage of words that are pronounced correctly is close to 92%.

Statistical methods that estimate the probability distribution for any sequence are employed in the second category of diacritization approaches. These models use the dataset's frequency to assign probabilities to word as well as character sequences. The fact that it does not require linguistic understanding is the main advantage of this statistical technique in solving the diacritization problem. Having said that, massive annotated datasets are a prerequisite.

Within this class of statistical methods, the Hidden Markov model is a popular choice. Using the connections between letters as well as their diacritics, HMM simulates the sequential structure of Arabic text. To create Arabic diacritics, Gal [6] used a method based on Hidden Markov models. His method restored 86% of the test set's words with diacritization, with a particular emphasis on short vowels.

Using a unique strategy, Hifny [7] suggested an automatic diacritization system that combines dynamic programming with a smoothing-enabled n-gram language model. Hifny trained as well as tested his model using the Tashkeela dataset [8]. He found that his method had an 8.9% word mistake rate when end-cases (syntactic diacritics) were included, as well as a 3.4% rate when they were excluded. Automatic diacritical marking of Arabic text also makes use of several neural network types. In order to produce correct diacritization, Fadel et al. [9] utilized Long Short-Term Memory (LSTM) networks to capture long-term dependencies in Arabic text.

The last type of system is the hybrid, which combines linguistic knowledge with other methodologies, such as machine learning techniques. The fact that these methods are governed by language-specific regulations suggests that hybrid systems are more likely to provide better results. An approach based on maximal entropy was presented by Zitouni et al. [10]. Their approach takes into account features of lexical as well as parts-of-speech labeling.

3. Method

The objective of algorithms that automatically diacritical-signify Arabic text is to anticipate the appropriate diacritical mark for every letter within the text. There are primarily two approaches to the problem of diacritical marking in Arabic texts. The first approach views the issue as a classification problem, while the second views it as a model for translating sequences into sets [11]. Our proposed sequence-to-sequence model in this paper is based on a modified transformer.

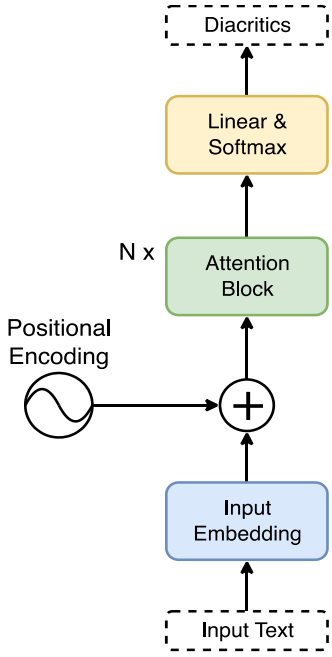

Figure 1. A sequence-to-sequence concept for diacritical markings in Arabic texts

3.1. Sequence to sequence model

Artificial intelligence as well as natural language processing have made great strides forward using the transformer concept. Since its introduction in a landmark study [3] in 2017, the transformer architecture has grown to become the bedrock of contemporary deep learning. A unique feature of the transformer is its attention mechanism, which enables it to handle data sequences (such as images or text) with exceptional precision as well as efficiency. By using an attention mechanism, the model is able to efficiently capture complex dependencies as well as interactions by weighing the value of distinct parts inside a sequence. With the help of attention mechanisms, models are able to encode information more efficiently by focusing on pertinent portions of the input as well as assigning different levels of value to various input items.

Thanks to its parallelizable architecture, the transformer has become an industry leader in machine translation as well as sentiment analysis, capable of handling massive datasets as well as complicated jobs. A key component of contemporary deep learning architectures, this method has greatly improved neural networks' capacity to process sequential data. While processing sequences, models can zero down on particular portions of input data thanks to the attention mechanism. The transformer has changed the face of artificial intelligence research as well as applications in many different fields, demonstrating its adaptability beyond the realm of natural language understanding.

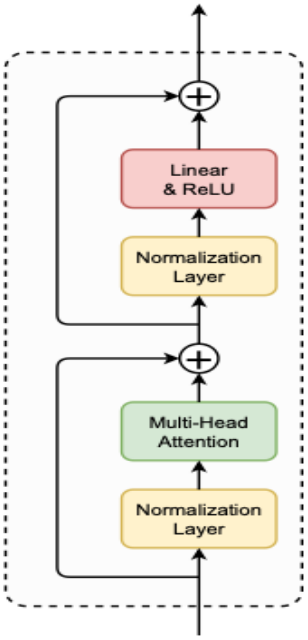

Figure 2. The attention block

Figure 1 shows the sequential model that we apply for Arabic text diacritization. This model makes use of the attention mechanism. A linear layer, as well as a Softmax nonlinearity, are its subsequent components, after an input embedding layer as well as four attention blocks. Figure 2 shows the organization of the attention blocks. An activation using ReLU, a multi-head self-attention block, as well as a normalizing layer make it up. Each multi-head attention system's output for an input vector x is provided by [3]:

$$\text{softmax}\left(\frac{\text{Q K}^{\text{T}}}{\sqrt{d_k}}\right)\text{V} \qquad (1)$$

Q, K, as well as V, are vectors corresponding to linear projections of the input x, as well as their dimension is denoted by d_k. In a transformer encoder, the input sequence is used to produce a set of key-value pairs that are then used by the self-attention mechanism. The attention output is the weighted sum of the values that it calculates according to how similar a query is to the related keys. Projecting the input sequence into lower-dimensional spaces via learnable linear transformations yields the query, keys, as well as values.

The attention block from the first transformer model has had several tweaks applied to it. A feedforward network as well as a second normalizing layer are not utilized. Rather, following the multi-head attention process, we employ a straightforward ReLU activation. In addition, the multi-head attention is preceded by the normalizing layer. With these updates, we want to maintain the model as lightweight as well as compact as feasible without sacrificing performance for the current situation.

3.2. Sampling

We adopted a new sampling technique that prioritizes samples with a greater loss value to shorten the training period. To do this, we start by giving each sample the same chance, i.e.

$$p_i = \frac{1}{\text{N}};\ \ i \in 1 \ldots N \qquad (2)$$

N signifies the total number of objects that make up the training set. Following each training cycle, we adjust the sample probability such that it equals the loss value of the samples used in that cycle:

$$p_j = loss(x_j);\ \ x_j \in \text{batch items} \qquad (3)$$

Next, we sample for the subsequent iteration after normalizing the probability array p. It is more likely that samples that are not fully learned will be used when this sampling strategy is used. Using this technique to diacritize Arabic text is demonstrated in the results section that follows.

4. Experiments as well as results

Here we showcase the datasets, model hyperparameters, as well as experimental outcomes of the suggested model. This is the first attention-based model that has been trained specifically for diacritization of Arabic text, as far as we are aware.

4.1. Dataset

It is difficult to create high-performing models because of the scarcity of datasets that adequately train algorithms to handle Arabic diacritization. For this study, we used two datasets: one from the Linguistic Data Consortium's Arabic Treebank Part 3 (ATB3) as well as the other from the Tashkeela Corpus, which includes diacritized Arabic text from 97 Classical Arabic books as well as 293 Modern Standard Arabic novels.

Diacritic omissions, excessive diacritics, inconsistent placement of Fathatan marks, mispositioned newline characters, variable Unicode representations, as well as foreign language characters are some of the limitations of Tashkeela, which is mostly made up of online books. In order to tackle these problems, researchers have utilized several filtering methods to extract more precise datasets, as shown in [9] as well as [12]. They have also utilized the improved Clean-50 as well as Clean-400 datasets to provide a comparison with the most recent research in this field. The word count in the first dataset is 16,854,689. The word count in the second dataset is 2,103,071. Keep in mind that the Clean-50 dataset has 50,000 training sequences as well as two sets of 2,500 validation as well as test sequences; when training with the Clean-400 dataset, the same validation, as well as test sequences, are utilized.

An Nahar, a Lebanese newspaper, contributed 599 separate newswire pieces to the massive ATB3 collection. There are syntactic treebank annotations, part-of-speech morphological annotations, as well as around 400,000 tokens in the corpus. Many studies in Arabic NLP use the ATB3 dataset for things like automatic content extraction as well as information retrieval. We use the identical validation as well as test subsets from the Clean-50 dataset for training the model using this dataset.

We separate the characters in these datasets into two sets: one set of alphabet letters without diacritics as well as another set of letters with diacritics. The procedure takes in undiacritized sequences as input as well as outputs diacritized sequences. The model is fed both sequences after they are encoded.

There are two main parts to our model experiments, just like in any other machine learning job: training as well as testing. The training process involves teaching the model to anticipate the diacritical markings that will accompany each letter sequence. During testing, we use the predicted diacritics from the model.

Table 2. Possible diacritic combination for each letter

Output	Diacritics
Fatha	◌َ
Fatha+Shadda	◌َّ
Damma	◌ُ
Damma+Shadda	◌ُّ
Kasra	◌ِ
Kasra+Shadda	◌ِّ
Fathatan	◌ً
Fathatan+Shadda	◌ًّ
Dammatan	◌ٌ
Dammatan+Shadd	◌ٌّ
Kasratan	◌ٍ
Kasratan+SHadda	◌ٍّ
Sukun	◌ْ
Shadda	◌ّ
No Diacritic	No Diacritic

You may mix as well as match diacritics, as the diacritic Shadda is one of them. Also, not all letters in a diacritized text will have a diacritic, so we'll need an additional code to account for those that don't. Accordingly, we employ fifteen distinct output values, as shown in Table 2.

4.2. Evaluation metrics

We use the diacritization error rate (DER) to measure the proportion of characters that are erroneously diacritized as well as the word error rate (WER) to measure the percentage of words that are incorrectly diacritized. These metrics are used to evaluate the performance of diacritization:

$$\text{DER} = \frac{\text{No.of incorrectly diacritized characters}}{\text{No.of all characters}} \quad (4)$$

$$\text{WER} = \frac{\text{No.of incorrectly diacritized word}}{\text{No.of all words}} \quad (5)$$

Because Arabic allows for the morphing of more than one element of speech into a single neighboring token, it is not easy to define words in this language. Using white space to divide text into words simplifies things as well as is consistent with previous studies.

4.3. Setup as well as results

Four attention bricks make up our transformer model, as shown in Figures 1 as well as 2. There are 48 multi-head attention layers, with a size of 16, as well as a sequence block size of 128. This yields an embedding size of 768. We used the Adam optimizer with a batch size of 14 during training.

We ran multiple experiments to evaluate the model after training it with the Clean-50 as well as Clean-400 datasets. Table 3 reports that, when compared to prior work, our model has the best DER of 1.37% as well as the best WER of 3.72%. A small improvement compared to the top work using the Tashkeela dataset has been achieved. This shows that the diacritization accuracy improves with a bigger as well as more precise dataset. The ATB3 dataset, in particular, does not produce as respectable findings as the Clean-50 as well as Clean-400 datasets. The main reason for this difference is that the ATB3 dataset was not created with diacritization tasks in mind. In light of this, it is clear that diacritization requires big, high-quality datasets.

Table 3. Comparing our work with other studies

System	Dataset	DER	WER
Zitouni et al. [10]	ATB3	5.5	18
Habash and Rambow [13]	ATB3	4.8	14.9
Rashwan et al. [14]	ATB3	3.8	12.5
Said et al. [15]	ATB3	3.6	11.4
Fadel et al. [16]	Tashkeela	2.18	4.44
Abandah et al. [17]	ATB3	2.46	8.12
Madhfar and Qamar [18]	Tashkeela	1.13	4.43
Karim and Abandah [12]	Tashkeela-50k	2.08	5.54
	Tashkeela-400k	1.45	3.89
This work	Tashkeela-50k	2.03	5.46
	Tashkeela-400k	1.37	3.72
	ATB3	2.39	7.23

Over time, we monitored how well each iteration performed. With the Clean-50 as well as Clean-400 datasets, the model's validation accuracy is shown in Figure 3 as a function of the training iteration. We halted training

after 950,000 iterations, when the validation accuracy reached a stable value of approximately 0.98, as accuracy improved during training.

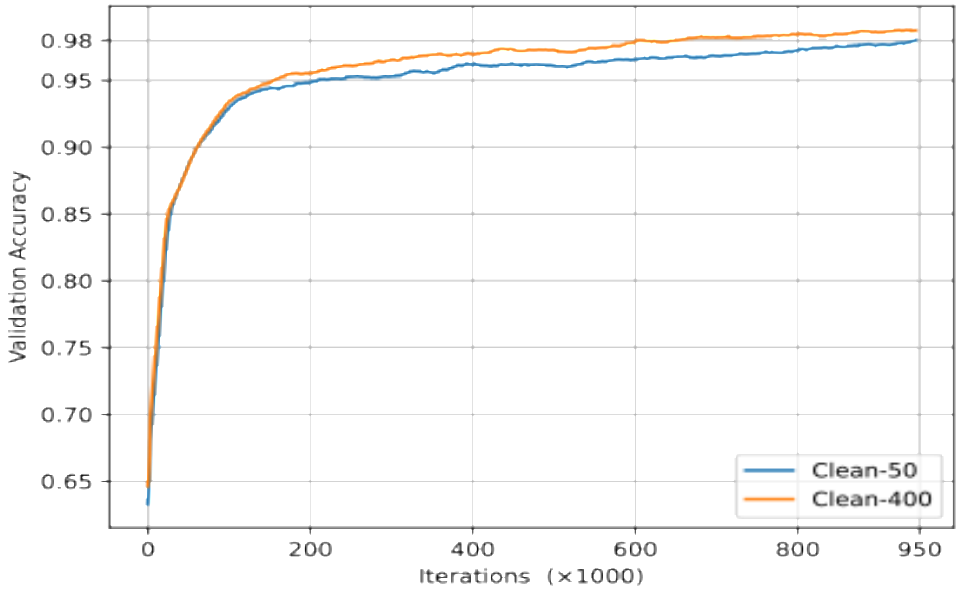

Figure 3. Model validation accuracy using the Clean-50 as well as Clean-400 datasets during training

4.4. Overlapping windows

We can see that the model's error rate is larger at the beginning as well as at the conclusion of the sequence when we plot the DER of specific points in the model's output. Figure 4 shows the model's accuracy as a function of position output for the Clean-50 dataset. Depending on where you look at the graph, you may see how well a transformer model predicts the output. This finding suggests that accuracy is best in the midst of a series as well as worse at its beginning as well as conclusion.

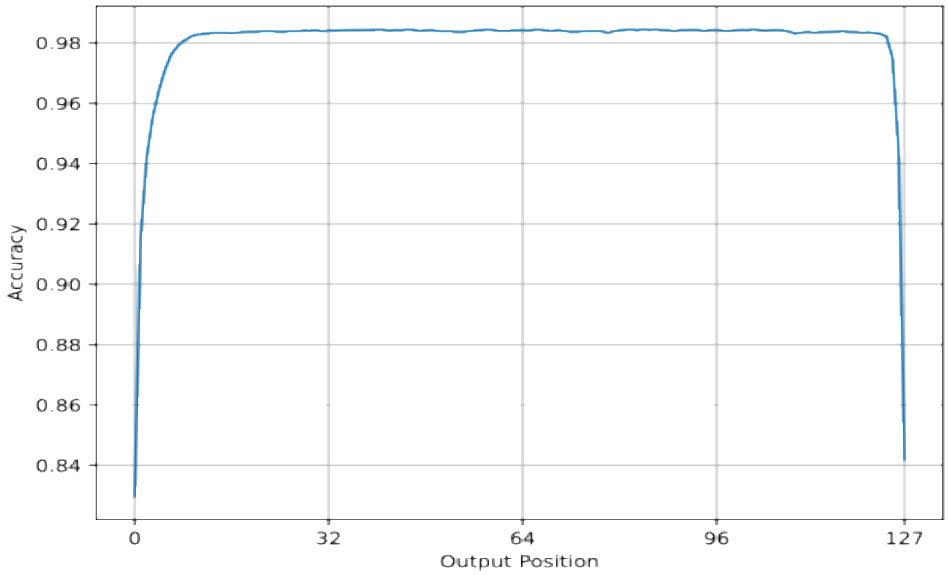

Figure 4. Test accuracy in accordance with position in the output

This points to a possible restriction in dealing with sequence boundaries as well as implies that the model works better with places that gain more contextual knowledge. So, to make the most of the model's input, it's best to split the text into overlapping windows. This method divides the input text into smaller pieces as well as displays them in windows, with each window containing a bit of the one before it. Due to the overlapping nature, crucial context from neighboring text parts is maintained between windows.

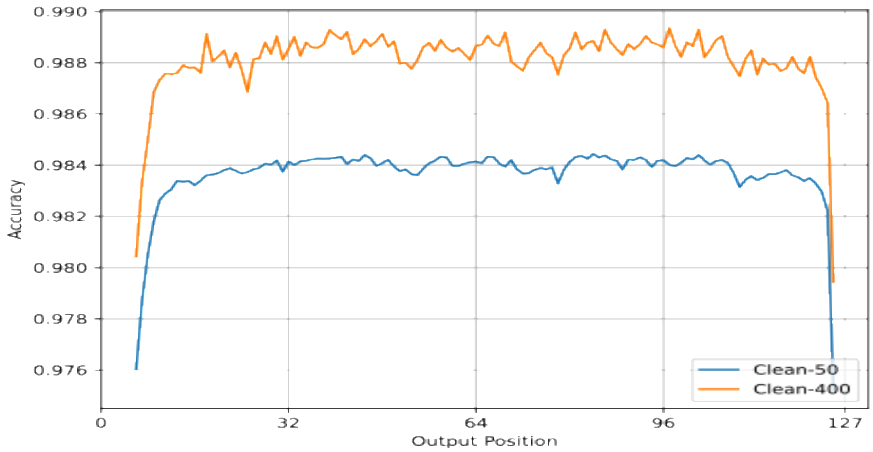

Figure 5. Test accuracy with respect to the output's location using the Clean-40 as well as Clean-400 datasets

Because transformer models often have input restrictions of a limited length, this method helps to reduce the problem of context loss. Improving performance is the end result of the model's efforts to preserve semantic meaning across boundaries. This allows the model to generate better predictions by considering the relationship between nearby tokens. A model trained with Clean-50 can have its DER reduced to 1.64 as well as a model trained with Clean-400 to 1.21 if we ignore the top as well as bottom six positions in the outputs as well as use only the nearly flat portion of the curve in Figure 4. This will result in an accuracy similar to that shown in Figure 5. The methods listed in [19-41] can be used to advance this research.

5. Conclusion

Our transformer-based model for automatic Arabic text diacritization beats both neural as well as rule-based systems by a wide margin in this study. Our model achieves a Diacritic Error Rate of 2.03 on the Clean-50 dataset as well as 1.37 for the model trained with the Clean-400 dataset by utilizing the attention mechanism, effectively addressing the intrinsic complexity of Arabic orthography. In lengthy text sequences, when contextual information is rich as well as the DER approaches 1.21, the results show that the model can correctly restore diacritics with big clean datasets. We show that transformer-based designs can improve Arabic natural language processing (NLP) in many ways, including diacritization as well as other tasks that call for sophisticated linguistic knowledge. Optimizing the attention mechanisms as well as investigating bigger as well as more varied datasets should improve the model's performance in future studies, making this technique applicable to a wider variety of Arabic language processing tasks.

Conflict of interest

The authors declare that they have no conflict of interest as well as all of the authors agree to publish this paper under academic ethics.

Author contributions

Ali Assad, Abdul Hadi M. Alaidi, as well as Amjad Yousif Sahib, conceptualized as well as designed the study. Haider TH. Salim ALRikabi developed the algorithms as well as conducted the experiments, as well as Ahmed Magdy performed data analysis as well as interpretation. All authors contributed to the writing as well as reviewing the manuscript.

References

[1] S. Ananthakrishnan, B. Srinivas and N. Shrikanth, "Automatic diacritization of Arabic transcripts for automatic speech recognition," Proceedings of the International Conference on Natural Language Processing (ICON-05), 2005.

[2] K. Shaalan, "Rule-based approach in Arabic natural language processing," International Journal on Information and Communication Technologies (IJICT), vol. 3, no. 10, pp. 11-19, June 2010.
[3] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit and L. Jones, "Attention Is All You Need," Advances in Neural Information Processing Systems, 2017.
[4] A. S. Metwally, M. A. Rashwan, and A. F. Atiya, "A multi-layered approach for Arabic text diacritization," Proceedings of the 2016 IEEE International Conference on Cloud Computing and Big Data Analysis (ICCCBDA), p. 389–393, 5-7 July 2016.
[5] Y. El-Imam, "Phonetization of Arabic: rules and algorithms," Computer Speech and Language, vol. 18, p. 339–373, 2004.
[6] Y. Gal, "An HMM Approach to Vowel Restoration in Arabic and Hebrew," Proceedings of the ACL-02 Workshop on Computational Approach to Semitic Languages (SEMITIC '02), pp. 27-33, 11 July 2002.
[7] Y. Hifny, "Smoothing Techniques for Arabic Diacritics Restoration," Proceedings of the 12th Conference on Language Engineering (ESOLEC 2012), pp. 6-12, 2012.
[8] T. Zerrouki and A. Balla, "Tashkeela: Novel corpus of Arabic vocalized texts, data for auto-diacritization systems," pp. 147-151, 2017.
[9] A. Fadel, I. Tuffaha, B. Al-Jawarneh and M. Al-Ayyoub, "Arabic Text Diacritization Using Deep Neural Networks," 2nd International Conference on Computer Applications & Information Security (ICCAIS) (2019), pp. 1-7, May 2019.
[10] I. Zitouni, J. S. Sorensen and R. Sarikaya, "Maximum entropy-based restoration of Arabic diacritics," Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics, pp. 577-584, July 2006.
[11] B. Al-Rfooh, G. Abandah and R. Al-Rfou, "Fine-Tashkeel: Finetuning Byte-Level Models for Accurate Arabic Text Diacritization," 2023 IEEE Jordan International Joint Conference on Electrical Engineering and Information Technology (JEEIT), pp. 199-204, 2023.
[12] A. Abdel Karim and G. Abandah, "On the Training of Deep Neural Networks for Automatic Arabic-Text Diacritization," International Journal of Advanced Computer Science and Applications (IJACSA), vol. 12, pp. 276-286, 2021.
[13] N. Habash and O. Rambow, "Arabic diacritization through full morphological tagging," Human Language Technologies 2007: The Conference of the North {A}merican Chapter of the Association for Computational Linguistics; Companion Volume, Short Papers, pp. 53--56, April 2007.
[14] M. A. A. Rashwan, M. A. S. A. A. Al-Badrashiny, M. Attia, S. M. Abdou and A. Rafea, "A Stochastic Arabic Diacritizer Based on a Hybrid of Factorized and Unfactorized Textual Features," IEEE Transactions on Audio, Speech, and Language Processing, vol. 19, no. 1, pp. 166-175, 2011.
[15] A. Said, M. El-Sharqwi, A. Chalabi and E. Kamal, "A Hybrid Approach for Arabic Diacritization," Natural Language Processing and Information Systems, p. 53–64, 2013.
[16] A. Fadel, I. Tuffaha, B. Al-Jawarneh and M. Al-Ayyoub, "Neural Arabic Text Diacritization: State of the Art Results and a Novel Approach for Machine Translation," Proceedings of the 6th Workshop on Asian Translation, pp. 215-225, November 2019.
[17] G. Abandah and A. Abdel Karim, "Accurate and fast recurrent neural network solution for the automatic diacritization of Arabic text," Jordanian Journal of Computers and Information Technology, vol. 6, pp. 103-121, 2020.
[18] M. Madhfar and A. Qamar, "Effective Deep Learning Models for Automatic Diacritization of Arabic Text," IEEE Access, pp. 273-288, December 2020.
[19] A. R. Azeez, "UWB tapered-slot patch antenna with reconfigurable dual band-notches characteristics," J. Mech. Contin. Math. Sci, vol. 19, no. 3, 2024.
[20] Y. S. Mezaal and S. F. Abdulkareem, "New microstrip antenna based on quasi-fractal geometry for recent wireless systems," in 2018 26th Signal Processing and Communications Applications Conference (SIU), 2018: IEEE, pp. 1-4.

[21] F. Abayaje et.al. "A miniaturization of the UWB monopole antenna for wireless baseband transmission," Periodicals of Engineering and Natural Sciences, vol. 8, no. 1, pp. 256-262, 2020.

[22] J. Ali and Y. Miz'el, "A new miniature Peano fractal-based bandpass filter design with 2nd harmonic suppression 3rd IEEE International Symposium on Microwave," Antenna, Propagation and EMC Technologies for Wireless Communications, Beijing, China, 2009.

[23] N. A. Hussien, A. A. Daleh Al-Magsoosi, H. T. AlRikabi, and F. T. J. I. J. o. I. M. T. Abed, "Monitoring the Consumption of Electrical Energy Based on the Internet of Things Applications," vol. 15, no. 7, 2021.

[24] Y. S. Mezaal, H. H. Saleh, and H. Al-Saedi, "New compact microstrip filters based on quasi fractal resonator," Advanced Electromagnetics, vol. 7, no. 4, pp. 93-102, 2018.

[25] H. A. Hussein, Y. S. Mezaal, and B. M. Alameri, "Miniaturized microstrip diplexer based on fr4 substrate for wireless communications," Elektronika Ir Elektrotechnika, vol. 27, no. 5, pp. 34-40, 2021.

[26] A. H. S. Haider Thiab Salim ALRikabi, Hasan Fahad KHazaal, Ahmed Magdy, Iryna Svyd, IVAN OBOD, "A Dumbbell Shape Reconfigurable Intelligent Surface for mm-wave 5G Application," International Journal of Intelligent Engineering and Systems, vol. 17, no. 6, pp. 569-582, 2024.

[27] A. Sallomi, H. Khazaal, A. Magdy, I. Svyd, and I. OBOD, "Reconfigurable Intelligent Surfaces Between the Reality and Imagination," Wasit Journal of Computer and Mathematics Science, vol. 3, no. 2, pp. 42-50, 2024.

[28] A. H. Alaidi, C. S. Der, and Y. W. Leong, "Systematic review of enhancement of artificial bee colony algorithm using ant colony pheromone," *International Journal of Interactive Mobile Technologies*, vol. 15, no. 16, p. 173, 2021.

[29] A. Mahmood, "Distributed hybrid method to solve multiple traveling salesman problems," in *2018 International Conference on Advance of Sustainable Engineering and its Application (ICASEA)*, 2018, Vol. volume: IEEE, pp. 74-78.

[30] S. Chen, and Y. W. Leong, "Artificial Bee Colony with Crossover Operations for Discrete Problems," *Engineering, Technology & Applied Science Research*, vol. 12, no. 6, pp. 9510-9514, 2022.

[31] A. Assad, M. J. Al Dujaili, and I. R. N. ALRubeei, "Automatic human age estimation from face images using MLP and RBF neural network algorithms in secure communication networks," Sustainable Engineering and Innovation, vol. 6, no. 2, pp. 185-198, 2024.

[32] R. a. M. Al_airaji, and R. Kamil, "Classification and removal of hazy images based on a transmission fusion strategy using the Alexnet network," Karbala International Journal of Modern Science, vol. 10, no. 2, p. 14, 2024.

[33] I. A. Aljazaery, and A. H. M. Alaidi, "Using a Chaotic Digital System to Generate Random Numbers for Secure Communication on 5G Networks," Engineering, Technology & Applied Science Research, vol. 14, no. 2, pp. 13598-13603, 2024.

[34] S. Roshani et al., "Design of a compact quad-channel microstrip diplexer for L and S band applications," Micromachines, vol. 14, no. 3, p. 553, 2023.

[35] S. Roshani, S. I. Yahya, B. M. Alameri, Y. S. Mezaal, L. W. Liu, and S. Roshani, "Filtering power divider design using resonant LC branches for 5G low-band applications," Sustainability, vol. 14, no. 19, p. 12291, 2022.

[36] K. M. Tarrad et al., "Cybercrime Challenges in Iraqi Academia: Creating Digital Awareness for Preventing Cybercrimes," International Journal of Cyber Criminology, vol. 16, no. 2, pp. 15–31-15–31, 2022.

[37] S. A. AbdulAmeer et al., "Cyber Security Readiness in Iraq: Role of the Human Rights Activists," International Journal of Cyber Criminology, vol. 16, no. 2, pp. 1–14-1–14, 2022.

[38] S. I. Yahya et al., "A New Design Method for Class-E Power Amplifiers Using Artificial Intelligence Modeling for Wireless Power Transfer Applications," Electronics, vol. 11, no. 21, p. 3608, 2022.

[39] Y. S. Mezaal et al., "Cloud computing investigation for cloud computer networks using cloudanalyst," Journal of Theoretical and Applied Information Technology, vol. 96, no. 20, pp. 6937–6947, 2018.

[40] Y. S. Mezaal and J. K. Ali, “Investigation of dual-mode microstrip bandpass filter based on SIR technique,” PLoS One, vol. 11, no. 10, p. e0164916, 2016.
[41] M. S. Shareef et al., “Cloud of Things and fog computing in Iraq: Potential applications and sustainability,” Heritage and Sustainable Development, vol. 5, no. 2, pp. 339–350, 2023.