2019 18th IEEE International Conference on Machine Learning and Applications (ICMLA)

Investigating Input and Output Units in Diacritic Restoration

Sawsan Alqahtani*‡ and Mona Diab †*
*The George Washington University
†AWS, Amazon AI
‡Princess Nourah Bint Abdulrahman University
sawsanq@gwu.edu and diabmona@amazon.com

***Abstract*—Diacritic restoration is the task of assigning diacritics (accents) for each character in a given segment. The typical input levels that have been previously used in diacritic restoration models are word and/or character units. In this paper, we investigate the use of subwords as input units along with their diacritic patterns (combinations of adjacent diacritics) as output, as an alternative to word or character-based models. Our experiments show that characters provide the optimal level of information for sequence-based diacritic restoration models across different languages. We additionally improved our diacritic restoration model by maximizing over the output diacritic sequence using a Conditional Random Field (CRF). Adding a CRF layer improves the performance on observed and unobserved words substantially for Arabic and marginally for Yoruba.**

***Index Terms*—Diacritic restoration, Subword Information, BiLSTM-CRF**

I. Introduction

Diacritic restoration is the task of assigning/restoring diacritics for each character in a given segment.[1] Diacritic restoration may be applied to text as an end task in itself or as a pre-processing step for other downstream applications such as text-to-speech and machine translation [1, 2, 3].

The typical input unit granularities that have been extensively used in diacritic restoration are words and/or characters. Theoretically, word level information better captures semantic and syntactic relationships in the sentence. However, word-level models suffer from sparsity due to insufficient training examples. Acquiring large training datasets that include all possible diacritic variants requires intensive time and effort. In addition, word-level models pose the challenge of computational complexity in training due to the large input and output vocabulary size. On the other hand, diacritic restoration at the character level encodes local contextual information, minimizing the sparsity issue and improving model generalization. However, character-level diacritic restoration models lose a level of semantic and syntactic information, increasing the possibility of the composition of invalid words in the test languages.

In this paper, we investigate two approaches to improve the overall performance in diacritic restoration models: (1) using subword units as input to the diacritic restoration models; (2) incorporating previously predicted diacritics in subsequent prediction steps. Both approaches attempt to avoid composing locally inconsistent outputs or invalid words. We experiment with three languages: Arabic, Vietnamese, and Yoruba.

In the first approach, the question that arises is whether subword units[2] could improve the overall performance of diacritic restoration. The ideal scenario is to balance the generalization capacity of character-based models with the semantic and syntactic consistencies observed in word level information. This question is motivated by the recent success of utilizing subword information in different Natural Language Processing (NLP) tasks such as word embeddings [4], automatic speech recognition [5], and machine translation [6].

To do so, we systematically analyze the impact of various inputs in improving the overall performance of sequence-based diacritic restoration. In particular, we vary the input representation (fixed and variable-size n-grams) and the output space such that each segment in the input has a corresponding output of the same length. Our intuition is that subword units could incorporate semantic and syntactic relationships while preserving open vocabulary to generalize the model beyond observed instances. However, our experiments yielded negative results, which supports characters as the optimal choice for input units for diacritic restoration in all test languages. For Vietnamese, one version of the variable-size n-gram based models show equal or slight increase in performance on observed words when compared to the character-based model. However, the improvement was not consistent across all versions of variable-size n-gram based models.

In the second approach, we investigate the impact of incorporating previously predicted diacritics in subsequent decisions by utilizing a CRF (Conditional Random Field) layer in our BiLSTM (Bidirectional Long Short term Memory) [7] architecture as described in [8]. The CRF objective tries to optimize the overall output sequence rather than each individual output, helping the model avoid locally inconsistent outputs. This allows the model to evaluate the complete diacritic sequence of the given input to match the sequence in human-annotated data. Our experiments show some improvement compared to the standalone BiLSTM sequence tagger in two

[1] Diacritical marks are placed above, below, or in-between letters to indicate pronunciation and may change the meaning of the composed words. Languages that include diacritics such as Arabic and Vietnamese often omit diacritics in the written text which results in an increase in lexical ambiguity.

[2] Any segment between character and word.

978-1-7281-4550-1/19/$31.00 ©2019 IEEE
DOI 10.1109/ICMLA.2019.00142

languages (Arabic and Yoruba), leading to state-of-the-art performance in diacritic restoration in Arabic.

II. Related Work

We focus our discussions on studies that address the problem of diacritic restoration by evaluating more effective classifiers or by incorporating different morphological and contextual features.

Subword Information:: Asahiah et al. [9] suggest that word-level models are preferred if sufficient resources are available. Otherwise, character-based models provide better solutions [10, 11, 12, 13, 14]. In general, prior studies dealt with sparsity at the word level using different approaches.

Some studies start with word-based diacritic restoration models and then back off to smaller input segments or other linguistic features if the word-based model does not provide acceptable solutions [11, 14, 15, 16, 17]. Other studies utilize subword information as input to the diacritic restoration model [18, 19, 20]. Examples of successfully applied subword units in diacritic restoration include linguistically motivated units such as morphemes [17, 21], syllables [20, 22], lemmas [23] as well as simple n-grams [19, 24]. For instance, Wagacha et al. [24] compared characters and character-based 3-grams in four languages, demonstrating that the use of characters has better performance in some languages than others.

Similar to our objective, Nguyen and Ock [19] conducted a systematic study to investigate the optimal input representation for modeling diacritic restoration in Vietnamese. They showed that character-based models provide the best performance when compared to the use of syllables, n-grams, or words as input to the diacritic restoration model. Our results also support this claim for three languages by using neural-based models and automatically learned features as compared to the classical machine learning algorithms shown in [19].

Sequence Taggers:: To address diacritic restoration, prior studies utilized classical machine learning algorithms such as Maximum Entropy [25], Hidden Markov Model [26], Conditional Random Fields (CRF) [12], and Support Vector Machines [27]. Recently, neural-based models that incorporate future and previous context (BiLSTM or A-Causal Temporal Convolutional Network (A-TCN)) provided state-of-the-art performance in diacritic restoration [13, 28, 29, 30, 31]. Even though both BiLSTM and A-TCN utilize bidirectional information, neither considers previously predicted diacritics when predicting the diacritics of the current segment. On the other hand, classical machine learning algorithms, such as CRF [32] or Maximum Entropy Markov models [33], consider previously predicted diacritics. Huang et al. [8] described an architecture based on BiLSTM-CRF to utilize automatically learned features while considering the previous predicted diacritics. We experiment with this architecture in diacritic restoration.

III. Approach

We trained neural-based diacritic restoration models (Section III-B) based on a given sequence of segmented words (Section III-A). The computational complexity between the models varied based on the input and output representations.

A. Input Representation

Representing sentences as characters or surface words (white space delimited) is self explanatory. The remaining input segments are represented by fixed and variable size n-grams. We chose this representation because they are easily extracted from the space tokenized text and more generalizable to new datasets. both approaches are applied to the same input space of words undiacritized and untokenized (white space delimited).

Fixed size n-grams:[3]*:* We converted the text into n-grams with stride equal to n (chunking).[4] For instance, the Arabic undiacritized word *wsyktb* وسيكتب "and he will write" is converted into the 3-gram sequence $\{$*wsy* وسي, *ktb* كتب$\}$. If the word length is not a multiple of n, the last segment will remain less than n (e.g. $\{$*syk* سيك, *tb* تب$\}$ for *syktb* "he will write"). We use $n = 2, 3, 4$.

Variable size n-grams:: We used Byte Pair Encoding (BPE) [6, 34], a technique that segments a word into variable-size sequences of characters in an iterative fashion, merging the most frequent pairs of character sequences with a unified and unique symbol. This technique is heavily used in different NLP applications, such as machine translation, to allow processing rare words from smaller segments. Each merge operation is constrained within the word boundaries and produces one new symbol. This symbol represents n-gram sequences of characters whose lengths vary from one merge operation to another. The generated vocabulary size must be equal to the number of unique characters in the dataset as well as the number of merge operations.[5] The text is then converted by matching the input text with the generated vocabulary.

TABLE I
Number of words for all languages.

Data	Arabic	Vietnamese	Yoruba
Train	503k	800k	800k
Test	63k	786k	44k
Dev	63k	408k	44k

B. Model Architecture

Bidirectional Long Short Term Memory (BiLSTM) [7] has shown great success for diacritic restoration in previous studies, leveraging long range dependencies and preserving the temporal order of the sequence. Thus, we use BiLSTM for sequential classification such that each input has a corresponding diacritic(s) of exactly the same length. For instance, the corresponding diacritics for $\{$*wsy* وسي, *ktb* كتب$\}$ would be $\{$*__w__a__s__a__y__a*, *__k__o__t__u__b__u*$\}$. For Arabic, including the attached

[3] Contiguous sequences of characters of fixed size equal to n.

[4] We apply n-gram chunking rather than a sliding window to have a dataset comparable to the variable size of n-grams.

[5] We tune the number of operations to include values within [100-30,000].

TABLE II
MODELS' PERFORMANCE ACROSS DIFFERENT LANGUAGES.[a]

Input	**Arabic**				**Yoruba**			**Vietnamese**		
	WER	**DER**	**OOV**	**Lexical WER**	**WER**	**DER**	**OOV**	**WER**	**DER**	**OOV**
state-of-the-art	**8.2**	-	**20.2**	-	**4.6**	-	-	4.5	1.8	-
character	**8.2**	**2.8**	*34.1*	**4.2**	*12.4*	**4.5**	**74.4**	3.3	***1.1***	***22.9***
2-grams	9.6	3.3	42.6	5.3	*12.4*	4.6	82.8	3.3	***1.1***	24.5
3-grams	9.9	3.7	50.3	5.6	12.9	5.2	91.6	5.0	1.7	36.4
4-grams	11.1	4.7	62.1	6.3	12.6	5.8	97.9	5.4	1.9	37.0
1k-bpe	9.9	5.9	42.9	5.5	12.9	4.8	81.9	***3.0***	***1.1***	23.1
5k-bpe	10.6	6.5	45.8	5.7	12.6	5.1	86.7	3.5	1.2	23.1
10k-bpe	10.9	6.6	48.8	6.0	12.8	6.4	91.8	3.5	1.2	23.6
word	15.0	9.2	86.4	9.5	27.1	14.8	87.4	4.4	1.6	24.9

[a] State-of-the-art performance indicates the best models' performance given the dataset and refers to [29]'s model for Arabic, [35]'s model for Vietnamese, and [30]'s model for Yoruba. All metrics are %. **Bold** numbers refer to the highest performance across models in each column. *Italic* numbers refer to the highest performance comparing the different input representation.

consonants (e.g. bold characters in the previous example) significantly increase the model complexity and reduce model performance. Thus, we only consider diacritics in the output vocabulary for Arabic[6] and both consonants and diacritics for Vietnamese and Yoruba which conform with the character-based diacritic restoration models in previous studies.

IV. EXPERIMENTAL SETUP

A. Dataset

For Arabic, we use parts 1, 2, and 3 of the Arabic Treebank (ATB) dataset, following the same data division as [36]. We use the datasets made available by [30] for Yoruba and by [35] for Vietnamese. We sample a moderate size subset of the training data for Vietnamese, roughly 3.7% to train the models. In the process, we remove all sentences that meet one of the following criteria from the training set: at least one word of more than 10 characters;[7] do not have at least one diacritic; contain more than 70 words or less than 5 words. We also replaced all numbers with a unified symbol. Table I shows statistics about the dataset we use for each language.[8]

We segment each sentence into space tokenized units; each unit is further segmented into its characters and passed through the model along with a specific number of previous and future words. We add the special word boundary "<w>" between units with a window size of 10.

B. Parameter Settings

We use Adam [37] for learning optimization with 0.001 for initial learning rate. We use 20 for the number of epochs, 300 for embedding size and 250 for hidden units in each direction. For regularization, we use 0.3 for dropout [38] and pick the model of the highest performance on the validation set. We use the default weight initialization in Keras Deep Learning library.[9]

TABLE III
DIACRITIZED AND UNDIACRITIZED OOV RATES FOR THE DIFFERENT VERSIONS OF SUBWORD DATASETS, SEPARATED BY / RESPECTIVELY.

Input	**Arabic**	**Yoruba**	**Vietnamese**
character	0 / 0	0 / 0	0 / 0
2-grams	0.2 / 0.01	0.02 / 0.02	0.02 / 0.01
3-grams	1.9 / 0.25	0.4 / 0.09	0.06 / 0.03
4-grams	4.2 / 1.73	1.1 / 0.37	0.18 / 0.12
1k-bpe	0.2 / 0	0.3 / 0.01	0.04 / 0.03
5k-bpe	0.9 / 0	0.9 / 0.03	0.05 / 0.03
10k-bpe	1.7 / 0	1.9 / 0.09	0.06 / 0.03
word	10.5 / 7.3	1.8 / 0.98	0.52 / 0.47

C. Evaluation Metrics

We use standard evaluation metrics for diacritic restoration: Word Error Rate (WER) and Diacritic Error Rate (DER), which are the percentages of incorrectly diacritized words and characters, respectively. Additionally, we compute WER on Out Of Vocabulary (OOV) words[10] to examine the models' ability to generalize beyond observed data. State-of-the-art models report their performance mainly using WER on all words and occasionally using the remaining metrics.

V. RESULTS AND ANALYSIS

A. Subword Evaluation

Table II shows the impact of subword units in the performance of diacritic restoration models in the test languages.

[6] For consistency, we add the symbol "e" for characters that do not have a corresponding diacritic(s) to hold its position in the diacritic sequence.

[7] Vietnamese is characterized by having short word length.

[8] The ratio between training and testing size is different across languages because we use available datasets by previous studies. In Vietnamese, we only sampled the training dataset but did not change the test or evaluation datasets for better comparison with previous studies.

[9] https://keras.io/

[10] Test word forms not observed during training .

Using our experimental set-ups, character is the optimal choice for Arabic and Yoruba across different evaluation metrics. As the model increases in vocabulary size as well as diacritic sets, the performance drops gradually reaching that of word-based models. We believe that this significant increase in input and output contributed greatly to the performance degradation.

Vietnamese deviates from this finding. The $1k$-bpe model provides better performance in terms of WER and equal performance in terms of DER. However, the remaining subword-based models do not show the same behavior; fixed-size n-gram models, in particular, drop in performance in two versions when compared to the word-based model. As opposed to Arabic and Yoruba, the word-based model in Vietnamese is an acceptable solution, close to the performance of character- or subword-based models. We believe that this led to the different observation in Vietnamese. Despite having small positive results from the $1k$-bpe model, though not significant, this improvement is inconsistent across experiments. Thus, we believe that the character-based model is the optimal choice for Vietnamese as well. In addition, the character-based model achieves the highest performance on unobserved words compared to the remaining models.

Lexical and Inflectional Diacritics in Arabic:: Diacritics in Arabic can be divided into lexical and inflectional. Lexical diacritics change both pronunciations and meanings of words. On the other hand, inflectional diacritics are added to adhere to the syntactic positions of words within the sentence, changing the words' pronunciations without their underlying meanings. Inflectional diacritics are incorporated by changing diacritics of the last character of the main word. For example, both "Ealam**a**" and "Ealam**u**" mean flag but they differ in the diacritic of the last character complying with their syntactic positions. Table II shows the Arabic models' performance when we consider lexical diacritics only (lexical WER). We observe that WER is always 50% lower which indicate that the original WER were largely attributed to inflectional diacritics even though they constitute a small fraction of overall diacritics. This shows the difficulty in predicting inflectional diacritics conforming with the same findings in prior studies.

That being said, we investigated whether the presence of inflectional diacritics negatively affects the learning of lexical diacritics. Inflectional diacritics require syntactic information for accurate prediction which we do not sufficiently provide in our model. Furthermore, the presence of inflectional diacritics increases the number of possible diacritic patterns for each segment. Thus, we modified the dataset to only include lexical diacritics by removing the diacritics from the last character of each word. We then trained the same set of models to investigate whether subword units provide better performance for lexical diacritics. However, we observed a similar performance as in lexical WER in Table II, supporting characters as the optimal input representation.

Subword Vocabulary OOV:: Table III shows the OOV rate for each model at the segment level for diacritized and undiacritized versions. By definition, BPE versions of the data create undiacritized segments that are highly frequent which led to covering more vocabulary in the test set. Although not severely impacted, we can observe that diacritized OOV rates are higher than their counterparts in the undiacritized versions which means that there are diacritic combinations in the test set not learned during training.

Qualitative Analysis:: We analyzed random examples of predicted words to get an intuition of performance degradation. Table IV shows examples for predicted words. We observed that infrequent diacritized segments yielded incorrect predictions while highly frequent segments mostly yielded correct predictions. Thus, we further measured the association between frequencies of diacritic patterns and the model performance to generalize beyond example words. In particular, we created three bins of words: words with high frequent segments, low frequent segments, and a mix of both high and low frequent segments. Then, we compared these predicted words with their counterparts in character-based models.

For Arabic, character-based models were consistently outperforming subword-based models across all bins. There were few cases in Yoruba where subword-models have better performance with high frequent segments but none of the differences were significant. This means that the association between frequencies of diacritized segments and models' performance is not affirmative. Thus, we further experimented with replacing low frequent segments into characters in BPE based versions of the datasets. Although further segmenting low frequent segments into characters increased the models' performance, neither of them beat the character based model performance. This indicates that there are other possible unidentified causes for performance degradation.

For Vietnamese, we compared the errors generated by the character-based model and the $1k$-bpe model to get an intuition behind the slight increase in performance. We did not observe errors that appear particularly in one model but not the other. In addition, the model is inconsistent in predicting some words, so the same diacritized words were incorrectly predicted in some situations but not the others. Furthermore, closely investigating random errors, we found that most correctly predicted words by the $1k$-bpe that were incorrectly predicted by the character-based models appeared as the full word rather than segmented into smaller units. This indicates that subword units can combine the benefits of the two worlds (character and word) but there are other unexplained variables that hinder the performance.

State-of-the-art Comparison:: The character-based model for Arabic shows equals performance in terms of WER to Zalmout and Habash [29]'s model; however, it yields significantly lower results for WER on OOV words. We believe that the performance of Zalmout and Habash [29]'s model on OOV words is better because they utilize language modeling in addition to other morphological features to choose the best diacritized form. For Yoruba, our models perform significantly worse than Orife [30]'s model which use sequence-to-sequence classification at the word level similar to that in

TABLE IV
EXAMPLES OF CORRECT AND INCORRECT PREDICTIONS FOR 2-GRAM AND 1K-BPE MODELS. | SEGMENTS THE WORD AND THEIR CORRESPONDING FREQUENCIES IN THE HUMAN ANNOTATED DATA ACCORDING TO THE MODEL.

Undiac Word	Correct Prediction	2-gram Prediction	1k-bpe Prediction
أسجل >sjl	أُسَجِّلُ >**usaj~ilu** I register [something]	أَسْجَلَ >**aso|jala** (5|2)	أَسْجُلُ >**aso|julu** (5|14)
يقصرون yqSrwn	يُقَصِّرُونَ **yuqaS~iruwna** they make something shorter	يُقَصِّرُونَ **yuqa|S~iru|wna** (92|3|1,671)	يَقْصِرُونَ **yaqo|Si|ru|wna** (9|154|1,296|1,873)

machine translation.[11] Furthermore, the use of sequence-to-sequence modeling for diacritic restoration in general yields diacritized sentences that are not of the same length as the input as well as words not present in the original sentence, making it an undesirable solution in the context of diacritic restoration. For Vietnamese, Naplava et al. [35] reports 2.45% for WER on a much larger dataset (∼25M words), which is significantly better than our model. However, we applied their model on the same subset that we used in our experiment, their model performs less by ∼1%. This is because we use different parameter settings and different dataset preparation than theirs explained in Section IV-A.

B. Adding CRF Layer

Table V shows the performance of character-based models with and without CRF as the top layer.[12] In terms of WER, we observe substantial improvement when we add CRF on the top layer in Arabic and marginal improvements when applied to Yoruba. In terms of WER on OOV words, we observe improvement by ∼2% in both Arabic and Yoruba. The situation with Vietnamese is different; the stand-alone BiLSTM tagger performs better than adding an additional CRF layer. When compared to the state-of-the-art performance, BiLSTM-CRF provides better performance than Zalmout and Habash [29]'s model in terms of WER but still lower in terms of WER on OOV words, justified by the same reasons we previously mentioned. We qualitatively analyzed the quality of predicted words with and without the CRF layer in Arabic. We did not notice types of errors that appear in one model but not the other, but rather the quantity of correctly predicted words changed. Even though using BiLSTM-CRF improved the performance, it introduced other types of errors that were not observed when we used the stand-alone BiLSTM.

TABLE V
BILSTM AND BILSTM-CRF MODELS' PERFORMANCE ACROSS TEST LANGUAGES.[a]

Model	Lang	WER	DER	OOV
BiLSTM	ar	8.2	2.8	34.1
	yo	12.4	**4.5**	74.4
	vi	**3.3**	**1.1**	**22.9**
BiLSTM-CRF	ar	**7.6**	**2.7**	**32.1**
	yo	**12.3**	**4.5**	**72.1**
	vi	3.6	1.2	23.3

[a] **Bold** numbers indicate higher or equal performance when compared to the same language using the other model.

VI. DISCUSSION & CONCLUSION

We investigated two approaches to improve the performance of diacritic restoration. For the first approach, we used subword units as input to the diacritic restoration models. However, this approach yielded negative results supporting characters as the optimal input representation across all test languages. This is opposed to what we originally expected in which subword units can better capture semantic and syntactic information in addition to the generalization ability that we typically observe when using characters. Although our explanation for performance degradation for subword units is not definitive, we believe that combinations of the discussed factors (e.g. increase in input and output vocabulary and OOV segments) contributed to such degradation.

For the second approach, we considered previously predicted diacritics when predicting the diacritic of the current character by using BiLSTM-CRF instead of BiLSTM. This showed marginal improvements on Yoruba and substantial improvements on Arabic. The use of CRF layer increased the performance on OOV words in particular. In general, we believe that the difference in behavior for Arabic and Yoruba in one side and Vietnamese in the other can be attributed to the number of undiacritized words that have more than one diacritic choice. Statistically speaking, only ∼4% of undiacritized words in Vietnamese may have more

[11] We also investigated neural-based sequence-to-sequence diacritic restoration for Arabic to learn the interactions between subword inputs as well as their corresponding diacritics. Our experiments show that training such architecture in Arabic is not stable and provides much lower performance than sequence classification. We believe that this is attributed to the nature of diacritics in Arabic where every character is expected to include a diacritic(s) as opposed to Vietnamese and Yoruba where diacritics are added to certain characters and words are characterized of having short length.

[12] We use BiLSTM-CRF only for character models as it provides the best input representation for all languages and does not require extensive computational resources which is increasingly demanding in BiLSTM-CRF. We also investigated the performance of BiLSTM-CRF using various levels of subword units. Although we faced some computational challenges due to larger output vocabularies (BiLSTM-CRF requires storing n*n matrix for the output layer, where n is the number of classes), we observed similar trends as BiLSTM with moderate vocabulary sizes; that is, characters were the optimal input units in terms of accuracy.

than one diacritized version whereas more than 20% of undiacritized words in Arabic and Yoruba are ambiguous in terms of diacritics.

Acknowledgment

We thank Hanan Aldarmaki and the anonymous reviewers for their valuable comments and suggestions.

References

[1] D. Vergyri and K. Kirchhoff, "Automatic diacritization of Arabic for acoustic modeling in speech recognition," in *COLING*, 2004.

[2] C. Ungurean, D. Burileanu, V. Popescu, C. Negrescu, and A. Dervis, "Automatic diacritic restoration for a TTS-based e-mail reader application," *UPB Scientific Bulletin*, 2008.

[3] S. Alqahtani, M. Ghoneim, and M. Diab, "Investigating the impact of various partial diacritization schemes on Arabic-English statistical machine translation," in *AMTA*, 2016.

[4] P. Bojanowski, E. Grave, A. Joulin, and T. Mikolov, "Enriching word vectors with subword information," *Transactions of the Association for Computational Linguistics*, 2016.

[5] V.-B. Le and L. Besacier, "Automatic speech recognition for under-resourced languages: application to Vietnamese language," *IEEE Transactions on Audio, Speech, and Language Processing*, 2009.

[6] R. Sennrich, B. Haddow, and A. Birch, "Neural machine translation of rare words with subword units," *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*, 2016.

[7] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, 1997.

[8] Z. Huang, W. Xu, and K. Yu, "Bidirectional LSTM-CRF models for sequence tagging," 2015.

[9] F. Asahiah, O. Odetunji, and E. Adagunodo, "A survey of diacritic restoration in abjad and alphabet writing systems," *Natural Language Engineering*, 2018.

[10] R. Mihalcea, "Diacritics restoration: Learning from letters versus learning from words," in *International Conference on Intelligent Text Processing and Computational Linguistics*, 2002.

[11] G. De Pauw, P. W. Wagacha, and G.-M. De Schryver, "Automatic diacritic restoration for resource-scarce languages," in *International Conference on Text, Speech and Dialogue*, 2007.

[12] M. T. Nguyen, Q. N. Nguyen, and H. P. Nguyen, "Vietnamese diacritics restoration as sequential tagging," in *Computing and Communication Technologies, Research, Innovation, and Vision for the Future (RIVF)*, 2012.

[13] Y. Belinkov and J. Glass, "Arabic diacritization with recurrent neural networks," in *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, 2015.

[14] K. Darwish, H. Mubarak, and A. Abdelali, "Arabic diacritization: Stats, rules, and hacks," in *Proceedings of the Third Arabic Natural Language Processing Workshop*, 2017.

[15] R. Nelken and S. M. Shieber, "Arabic diacritization using weighted finite-state transducers," in *Proceedings of the ACL Workshop on Computational Approaches to Semitic Languages*, 2005.

[16] T. Schlippe, T. Nguyen, and S. Vogel, "Diacritization as a machine translation problem and as a sequence labeling problem," in *Proceedings of the Eighth Conference of the Association for Machine Translation in the Americas (AMTA)*, 2008.

[17] A. Said, M. El-Sharqwi, A. Chalabi, and E. Kamal, "A hybrid approach for Arabic diacritization," in *International Conference on Application of Natural Language to Information Systems*, 2013.

[18] T. Truyen, P. Dinh, and S. Venkatesh, "Constrained sequence classification for lexical disambiguation," in *Pacific Rim International Conference on Artificial Intelligence*, 2008.

[19] K.-H. Nguyen and C.-Y. Ock, "Diacritics restoration in Vietnamese: letter based vs. syllable based model," in *Pacific Rim International Conference on Artificial Intelligence*, 2010.

[20] F. Asahiah, O. Odejobi, and E. Adagunodo, "Restoring tone-marks in standard Yorùbá electronic text: improved model," 2017.

[21] D. Tufis and A. Chitu, "Automatic diacritics insertion in Romanian texts," in *Proceedings of the International Conference on Computational Lexicography COMPLEX*, 1999.

[22] L. Liu and D. Nouvel, "A Bambara tonalization system for word sense disambiguation using differential coding, segmentation and edit operation filtering," in *Proceedings of the Eighth International Joint Conference on Natural Language Processing*, 2017.

[23] J. Kanis and L. Müller, "Using lemmatization technique for automatic diacritics restoration," *SPECOM proceedings*, 2005.

[24] P. Wagacha, G. De Pauw, and P. Githinji, "A grapheme-based approach for accent restoration in Gikuyu," in *Proceedings of the Fifth International Conference on Language Resources and Evaluation (LREC)*, May 2006.

[25] I. Zitouni and R. Sarikaya, "Arabic diacritic restoration approach based on maximum entropy models," *Computer Speech & Language*, 2009.

[26] Y. Gal, "An HMM approach to vowel restoration in Arabic and Hebrew," in *Proceedings of the ACL workshop on Computational approaches to semitic languages*. Association for Computational Linguistics, 2002.

[27] K. Shaalan, H. Abo Bakr, and I. Ziedan, "A hybrid approach for building Arabic diacritizer," in *Proceedings of the EACL 2009 workshop on computational approaches to semitic languages*, 2009.

[28] G. A. Abandah, A. Graves, B. Al-Shagoor, A. Arabiyat,

F. Jamour, and M. Al-Taee, “Automatic diacritization of arabic text using recurrent neural networks,” *International Journal on Document Analysis and Recognition (IJDAR)*, 2015.

[29] N. Zalmout and N. Habash, “Don’t throw those morphological analyzers away just yet: Neural morphological disambiguation for Arabic,” in *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, 2017.

[30] I. Orife, “Attentive sequence-to-sequence learning for diacritic restoration of Yorùbá language text,” *Interspeech*, 2018.

[31] S. Alqahtani, A. Mishra, and M. Diab, “Convolutional neural networks for diacritic restoration,” in *EMNLP*, 2019.

[32] J. Lafferty, A. McCallum, and F. Pereira, “Conditional random fields: Probabilistic models for segmenting and labeling sequence data,” in *Proceedings of the Eighteenth International Conference on Machine Learning*, 2001.

[33] A. McCallum, D. Freitag, and F. Pereira, “Maximum entropy markov models for information extraction and segmentation.” in *ICML*, 2000.

[34] P. Gage, “A new algorithm for data compression,” *C Users J.*, 1994.

[35] J. Naplava, M. Straka, P. Stranak, and J. Hajic, “Diacritics restoration using neural networks,” in *Proceedings of the 11th Language Resources and Evaluation Conference*, 2018.

[36] M. Diab, N. Habash, O. Rambow, and R. Roth, “LDC Arabic treebanks and associated corpora: Data divisions manual,” 2013.

[37] D. Kingma and J. Ba, “Adam: A method for stochastic optimization,” 2014.

[38] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, “Dropout: a simple way to prevent neural networks from overfitting,” *The Journal of Machine Learning Research*, 2014.