Automatic Diacritization as Prerequisite Towards the Automatic Generation of Arabic Lexical Recognition Tests

Osama Hamed
Language Technology Lab
University of Duisburg-Essen
osama.hamed@uni-due.de

Torsten Zesch
Language Technology Lab
University of Duisburg-Essen
torsten.zesch@uni-due.de

Abstract

The automatic generation of Arabic lexical recognition tests entails several NLP challenges, including corpus linguistics, automatic diacritization, lemmatization and language modeling. Here, we only address the problem of automatic diacritization, a step that paves the road for the automatic generation of Arabic LRTs. We conduct a comparative study between the available tools for diacritization (Farasa and Madamira) and a strong baseline. We evaluate the error rates for these systems using a set of publicly available (almost) fully diacritized corpora, but in a relaxed evaluation mode to ensure fair comparison. Farasa outperforms Madamira and the baseline under all conditions.

1 Introduction

Lexical recognition tests are widely used to assess vocabulary knowledge. LRTs are based on the assumption that recognizing a word is sufficient for 'knowing' the word (Cameron, 2002). In such tests, the participants are being shown a list of items, containing words and nonwords. Their task is based on word recognition approach, i.e. they have to say 'Yes' when the item is word and 'No' otherwise – see Figure 1.

In the past LRTs were manually generated, as in LexTALE[1] and other LexTALE-like tests (Lemhöfer and Broersma, 2012). However, for the repetitive testing as used in formative assessment (Wang, 2007), LRT's test stimuli need to be generated automatically using natural language processing (NLP) techniques. The automatic generation of LRTs involves two NLP tasks: (i) a simple task: words selection from a corpus, and (ii) a complex task: nonwords generation. Some researchers have recently proposed an approach to generate nonwords automatically using character n-gram language models as obtained from Brown corpus (Hamed and Zesch). They applied their approach to English, and considered word selection using frequency per million word.

We want to generalize their approach to other languages, and more specifically Arabic, which is both interesting and challenging language. Creating Arabic lexical recognition tests is a task that entails a lot of NLP challenges regarding automatic diacritization, corpus linguistic, morphological analysis e.g. lemmatization and language modeling.

While there exist well-established lexical recognition tests for English, and other European languages like German and Dutch (Lemhöfer and Broersma, 2012), French and Spanish, for many under-resourced languages, like Arabic, a lot of challenges still remain. We are aware of very few studies for Arabic, like (Ricks, 2015; Baharudin et al., 2014). Both studies were conducted without any diacritical marks, which means that the respondent claims to know the most frequent diacritized form of a word. Although some researchers have recently shown that the diacritical marks play a vital role in improving the difficulty of Arabic LRTs (Hamed and Zesch, 2017), they did not automate the whole process.

In this paper, we address one of these NLP challenges by taking a closer look on the different approaches for Arabic automatic diacritization, a dominant step in the design process of Arabic tests and especially the role of lexical diacritics that are a defining feature of Arabic word sense. Next, we provide some background on lexical recognition tests, followed by the entailed NLP challenges.

[1] The Lexical Test for Advanced Learners of English

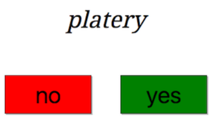
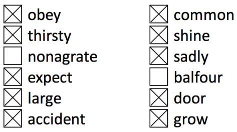


Figure 1: Example of a lexical recognition test as Yes/No question.



☒ obey
☒ thirsty
□ nonagrate
☒ expect
☒ large
☒ accident

☒ common
☒ shine
☒ sadly
□ balfour
☒ door
☒ grow

Figure 2: Example of a lexical recognition test in checklist format.

2 Related Work

The lexical recognition tests are typically used to measure the size of vocabulary, i.e. they only measure the breadth of vocabulary knowledge, but not the depth or quality (Schmitt, 2014). As described by Read (2004), breadth is used to "refer to any vocabulary measure that requires just a single response to each target word, by indicating whether the word is known or not". LRTs have two presentation formats: Yes/No question format, or checklist format – as show in Figures 1, 2.

2.1 Arabic LRTs

We are aware of a limited set of studies on Arabic lexical recognition tests.

– Test of Arabic Vocabulary (TAV) Baharudin et al. (2014) developed the *Test of Arabic Vocabulary* that uses 40 words selected from a book by panel of experts, but no nonwords. Thus, the test is vulnerable to test-wiseness or overconfidence (just answering 'yes' for each item).

– Test of Arabic Checklist Ricks (2015) developed a checklist-format test with 40 words and 20 nonwords (following the format introduced with LexTALE). Words were randomly selected from the Buckwalter/Parkinson frequency dictionary (Buckwalter and Parkinson, 2014), but excluding dialectal words. Nonwords were manually created using letters substitution approach.

Importance of Diacritics for LRTs In a recent studies (Hamed and Zesch, 2017, 2018), the researchers added a new parameter to Ricks's test. They constructed a diacritized test, where they partially diacritized the test stimuli (words and nonwords) and applied a form of relaxation that drops some diacritics. It was shown that diacritics play a vital role in words recognition, especially for beginner and intermediate learners. Hamed and Zesch (2018) demonstrated the impact of diacritization on increasing the difficulty of Arabic LRTs.

3 NLP Challenges

Three Arabic NLP challenges are entailed in the automatic generation of Arabic LRTs.

3.1 Diacritized Text Availability

LRT is a corpus-based assessment. To obtain a reliable frequency count, we typically need a large set of diacritized text. However, the currently available diacritized corpora are limited to religion related texts (Classical Arabic) such as the Holy Quran[2], RDI[3] and Tashkeela (Zerrouki and Balla, 2017) or newswire genres available in Penn Arabic Treebanks (PATB) from the Linguistic Data Consortium (LDC). Below we shed the light on the limitations of available corpora.

- **Religious Text** As we are trying to build education application that measure language proficiency. We need text that cover a variety of themes like: politics, economics, health, science and technology, sports, arts, culture and religion.

- **ATB** Which is limited in terms of size with less than 570k tokens and in terms of diversity with 87,160 unique surface forms (excluding numerals). In comparison, the AFP news corpus has approximately 765,890 unique tokens (Cole et al., 2001). Moreover, ATB often uses inconsistent diacritizations (Darwish et al., 2017).

Such a huge corpus can be crawled from the internet, lemmatized and diacritized accordingly.

3.2 Lemmatization

Lemmatization is the process of grouping together the different inflected forms of a word, so they can be analyzed as a single item (a.k.a the lemma). The lemma (aka the dictionary citation form) is a conventionalized choice using one of the word forms to stand for the set (Habash, 2010). Typically, the lemmas are written without any clitics

[2] http://tanzil.net/download/
[3] http://www.rdi-eg.com/RDI/TrainingData/

and without any sense (meaning indices). For example, the lemma of a verb is the third person masculine singular perfective form e.g. اتصل /AtSl/; while the lemma for a noun is the masculine singular form e.g. بيت /bayt/.[4]

Lemmas are usually presented in the LRTs for English and other European languages . Following standard practice for frequency lists in English and other languages, Buckwalter and Parkinson (2014) adopt the lemma as the organizing principle in their frequency dictionary. Lemmatization is difficult because Arabic is a morphology rich language and its words are highly inflected and derived (Aqel et al., 2015). We are aware of a well-established research that compares the available lemmatization tools. For example, Darwish and Mubarak (2016) have shown that Farasa outperforms or equalizes state-of-the-art Arabic segmenters like Madamira (Pasha et al., 2014). Next, we investigate the performance of Arabic diacritization tools.

3.3 Diacritization

The Arabic script contains two classes of symbols for writing words: letters and diacritics (Habash, 2010). Diab et al. (2007) grouped the diacritical marks into three categories: vowels (Fatha /a/, Damma /u/, Kasra /i/ and Sukun to indicate the no presence of any vowel), nunations or Tanween (Fathatan, Dammatan, Kasratan) and Shadda (gemination or a consonant doubling). The following examples show the appearance of all diacritics on the Arabic letter "د" /d/ grouped by categories: short vowels (دَ /da/, Damma دُ /du/, Kasra دِ /di/ and Sukun دْ /do/), Tanween (دًا /daN/, دٌ /duN/ and دٍ /diN/) and Shadda (دّ /dd/) respectively. Diacritization is the task of restoring missing diacritics automatically in languages that are usually written without diacritics like Arabic and Hebrew. We are not going to reinvent the wheel, instead we are going to evaluate the existing and freely available diacritization tools and report the best performing one. We are aware of two tools: MADAMIRA and Farasa. Next, we provide some background on Arabic automatic diacritization.

Corpus	Description	Availability	# of words
Quran	Religious	Free	78 K
RDI	Religious	Free	20 M
Tashkeela	Religious	Free	60 M
PATB	News	Commercial	1 M

Table 1: Summary of diacritized corpora.

4 State of the Art Overview

We shortly describe the diacritized datasets, and give an overview of the results that have so far been obtained on different corpora using the standard evaluation metrics.

4.1 Datasets

Table 1 summarizes the existing diacritized corpora, we conduct our experiments using Free corpora.

4.2 Evaluation Metrics

Two standard evaluation metrics are used almost exclusively to measure the system performance, in terms of error rates on the character and word levels. Namely, diacritization error rate (DER) and word error rate (WER). The smaller the error rates, the better the performance.

Case Endings In the diacritized version of the LRTs, the test stimuli are typically shown with lexical diacritics and without syntactic diacritics (a.k.a. case endings). Thus, we are going to report a variant of the above two mentioned metrics that ignore the word's last letter, denoted as **DER-1** and **WER-1**.

4.3 Results Overview

The existing diacritization approaches can be grouped into four main categories: statistical, sequence labeling, morphological analysis, and hybrid approaches (Metwally et al., 2016). Although there are several models within each approach, we only shed the light on one tool (not necessarily the the best performing one). A detailed review can be found in (Azmi and Almajed, 2015).

Statistical For example, the approach by Hifny (2012) is using an n-gram language model.

Sequence Labeling Some researchers have proposed handling the problem as a sequence labeling problem in which every letter of the word may be

[4] If no masculine is possible, then the feminine singular.

tagged with any of the possible diacritics. For example, recurrent neural networks model (Abandah et al., 2015).

Morphological Analysis For example, the system by Habash et al. (2009) is based on MADA, the tool for the morphological analysis and disambiguation of Arabic.

Hybrid Usually, the hybrid approach combines multiple-layers, each is utilizing one single approach. For example, the Rashwan et al. (2011) combines the unfactorized system (dictionary-based system) and a morphological analyzer.

Table 2 gives an overview of the reported results from the literature. The results are grouped by the corpus that was used for testing in order to allow for a fair comparison. Most numbers are still not directly comparable as they were obtained using different test sets. As most of the systems from the literature are not freely available, we have no way of directly comparing them. In this paper, we establish a comparative study that only includes the systems that are freely available along with the freely available corpora under a controlled settings.

5 Experimental Setup

The experiments are carried out using DKPro TC, the open-source UIMA-based framework for supervised text classification (Daxenberger et al., 2014). All the experiments were conducted as tenfold (1 part testing, 9 parts training) cross validation reporting the average over the ten folds.

5.1 Used Data

Because of the commercial availability of LDC's PATB datasets, our experimental data are drawn from the Quran, Tashkeela (CA) and RDI (contemporary writing). Table 3 shows the statistics for these three sub-datasets (punctuation marks are not counted).

Data Preprocessing The files from Tashkeela and RDI contain Quranic symbols or English alphabets and numerics respectively. In order to prepare them for training and testing purposes, the following preprocessing steps are performed: (i) convert them from HTML to plain text files that have one sentence per line. (ii) clean the files by removing the Quranic symbols and words written in non Arabic letters. (iii) normalize the Arabic text by removing the extra white spaces and the Tatweel.

5.2 Sequence Labeling Baseline

This treats diacritization as a sequence labeling (multi-class text classification) problem and proposed a baseline solution using conditional random fields (Lafferty et al., 2001).

Given a sentence (set of non-diacritized words) separated using white-space delimiter, each word in the sentence is a sequence of characters, and we want to label each letter with its corresponding labels from the diacritics set D = (d_1, ..., d_N). We represent each word as input sequence X = (x_1, ..., x_N), where we need to label each consonant in X with the diacritics that follow this consonant. Thus, the diacritization of X sequence is to find its labeling sequence Y, of word length and derived from D. A word might have more than one valid labeling. For the word "ktAb" (كتاب) X = (k, t, A, b), Y_1 = (i, a, o, u) and Y_2 = (i, a, o, a) are examples of two possible labeling.

Our features are character n-grams language model (LM) in sequence labeling approach. The features extractor selects the character-level features relevant to diacritics from annotated corpora. It collects the diacritics on previous, current and following character and up to the 6th character.

5.3 Diacritization Tools

We are aware of a few tools that can be tested with thousands of words, enhanced or integrated with Java Frameworks.

MADAMIRA A fast, comprehensive tool for morphological analysis and disambiguation of Arabic (Pasha et al., 2014). It is the successor of MADA (Habash et al., 2009).

Our experiments are carried out using the SAMA enabled version of Madamira *v2.1*. Madamira reported the accuracy of 86.3 and 95.3 for full and partial diacritization using an MSA blind test set. Madamira was used to diacritize the test sequences from the three corpora: Quran, Tashkeela (CA) and RDI. As the resulting diacritized text is encoded using Buckwalter transliteration, it is necessary to decode it into Arabic text. We compare the mapped Arabic text with gold sequence and calculating the different evaluation metrics.

Farasa A fast and accurate text processing toolkit for Arabic text (Darwish and Mubarak,

Corpus	Test Size (10^3)	Approach	All Diacritics		Ignore Last	
			DER	WER	DER-1	WER-1
ATB (Parts 1–3)	52	Morphological (Habash et al., 2009)	4.8	14.9	2.2	5.5
	52	Hybrid (Rashwan et al., 2011)	3.8	12.5	**1.2**	**3.1**
	37	RNN (Abandah et al., 2015)	**2.7**	**9.1**	1.4	4.3
Quran	76	RNN (Abandah et al., 2015)	**3.0**	**8.7**	**2.0**	**5.8**
Tashkeela	1902	Statistical (Hifny, 2012)	-	8.9	-	**3.4**
	272	RNN (Abandah et al., 2015)	**2.1**	**5.8**	**1.3**	3.5
Tashkeela+RDI	199	Hybrid (Bebah et al., 2014)	7.4	21.1	3.8	7.4

Table 2: List of Arabic Diacritization Systems.

ID	Corpus	# words (10^3)	∅ chars per word	Words / sentence
Q	Quran	78	4.25	12.6
R	RDI	297	4.47	34.1
T	Tashkeela	4,926	4.11	14.7

Table 3: Statistics of corpora sub-datasets used in this study.

2016).

We did not find any reported published results for Farasa diacritizer. We use Farasa to diacritize test sequence from the three corpora. We compare the resulting diacritized text with gold sequence and calculate the different metrics.

5.4 Evaluation Metrics

The evaluation was conducted across the character and word levels. For the Arabic LRTs, the test stimuli are not fully diacritized, instead they conform to specific diacritization (no default diacritics, no case-endings) settings. Thus, the different error rates are reported in relaxed mode, not in strict mode:

Strict Mode Whenever a letter has a set of diacritics in the gold standard, the tools are expected to predict exactly this set. This means that we punish tools that only provide a partial diacritization, e.g. by not returning some default diacritics. For example (قال) /qAl/ instead of (قَال) /qaAl/.

Relaxed Mode Whenever a letter has a set of diacritics in the gold standard, we do not expect the tool to predict exactly this set. Which means that we do not punish the tools on a letter that does not hold a diacritic. Instead, we only count for the letters that holds diacritic. This assumption remains valid only for words that are labeled with at least one diacritic by the diacritization tool (i.e. the tool is punished if no-diacritics are provided).

The following pre/post-processing steps are applied on the text to do the comparison in relaxed mode.

- **Comply to Default Diacritics** It is important to note that both Madamira and Farasa ignore the default diacritics, so that we normalize the gold sequence in such a way that also ignores the default diacritics to ensure fair comparison.
- **Sukun Removal** Some writing styles use the Sukun diacritic to mark un-diacritized letters and some styles leave such letters without any diacritic. To overcome these differences when computing the error rates, we discard the Sukun to neglect it in our evaluation.

5.5 Making Results Comparable

In Table 4, we show the average number and ratio of diacritics per letter for the gold standard and all systems used in our experiments. It shows that Madamira and Farasa both assign about the same amount of diacritics on average, but substantially fewer than the gold standard. This means that both tools are especially punished by the strict evaluation. These findings motivate us to do the evaluation using the *relaxed mode*. This requires us to normalize the ratio of letters with diacritics in the gold standard, training and output texts.

Table 5 shows the results in relaxed mode. The error rates are generally as expected. Farasa diacritizer outperforms all other methods in all conditions. The performance of Madamira with the Quran is lower than its performance with RDI and

	Avg.			Ratio		
Approach	Quran	RDI	Tashkeela	Quran	RDI	Tashkeela
Gold	.84	.83	.83	.78	.77	.77
Seq. Labeling	.82	.78	.78	.77	.74	.74
Madamira	.55	.59	.61	.51	.54	.56
Farasa	.58	.58	.61	.55	.54	.58

Table 4: Average number of diacritics per letter (Avg.), and the ratio of letters with diacritics (Ratio).

Tashkeela, and it outperforms the baseline with RDI and Tashkeela under all conditions. Farasa gets its best WER with RDI corpus, and outperforms almost at the same levels with Quran and Tashkeela. Madamira also performs almost on the same level with RDI and Tashkeela.

As most of the systems from the literature are not freely available, we have no way of directly comparing our results with those approaches unless they have the same settings. Only Farasa comes closer to the DER and DER-1 numbers by (Abandah et al., 2015) in Table 2 when text is drawn from the Quran. If we ignore the sample size, it can be clearly seen that the results of Farasa in relaxed mode are on the same level under (DER and DER-1) and outperforms the results approached by Bebah et al. (2014) under (WER and WER-1). The error rates are relatively high, we expect a certain level of overfitting on the domain (due to free words order) to play a role and that our results are closer to the actual performance that can be expected from existing tools.

Recall that Madamira reported an accuracy of 86.3% when evaluated using a blind MSA test set from the PATB. Madamira performs better in the relaxed mode (there is a slight difference). For instance, on average it shows a 74%, 80% and 80% WERs with the Quran, RDI and Tashkeela respectively. On the other hand, Farasa reported an accuracy of 86% with the three corpora.

6 Conclusion and Future Work

Arabic LRTs are corpus-based assessments that make use of diacritized words counts in a huge corpus. The lack of diacritized Arabic resources is one of the main challenges entailed in the automatic generation of Arabic LRTs. This paper approached the lack of diacritized Arabic resources via automatic diacritization. We presented a comparative study between the publicly available tools for diacritization. The evaluation experiments are conducted using diacritized text from the Quran, Tashkeela and RDI corpora, but in a relaxed evaluation mode to ensure fair comparison and suit the design of Arabic LRTs. Farasa outperforms Madamira under all conditions. In future work, we want to investigate the creation of dialectal Arabic lexical recognition tests automatically.

References

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

Afnan Aqel, Sahar Alwadei, and Mohammad Dahab. 2015. Building an Arabic Words Generator. *International Journal of Computer Applications*, 112(14).

Aqil Azmi and Reham Almajed. 2015. A survey of automatic arabic diacritization techniques. *Natural Language Engineering*, 21(03):477–495.

Harun Baharudin, Zawawi Ismail, Adelina Asmawi, and Normala Baharuddin. 2014. TAV of Arabic language measurement. *Mediterranean Journal of Social Sciences*, 5(20):2402.

Mohamed Bebah, Chennoufi Amine, Mazroui Azzeddine, and Lakhouaja Abdelhak. 2014. Hybrid approaches for automatic vowelization of arabic texts. *arXiv preprint arXiv:1410.2646*.

Tim Buckwalter and Dilworth Parkinson. 2014. *A frequency dictionary of Arabic: Core vocabulary for learners*. Routledge.

Lynne Cameron. 2002. Measuring vocabulary size in English as an additional language. *Language Teaching Research*, 6(2):145–173.

Andy Cole, David Graff, and Kevin Walker. 2001. Arabic newswire part 1 corpus (1-58563-190-6). *Linguistic Data Consortium (LDC)*.

Kareem Darwish and Hamdy Mubarak. 2016. Farasa: A New Fast and Accurate Arabic Word Segmenter. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC 2016)*, Paris, France. European Language Resources Association (ELRA).

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.

Johannes Daxenberger, Oliver Ferschke, Iryna Gurevych, Torsten Zesch, et al. 2014. DKPro TC: A Java-based Framework for Supervised Learning Experiments on Textual Data. In *ACL (System Demonstrations)*, pages 61–66.

Corpus	Approach	All Diacritics		Ignore Last	
		DER	**WER**	**DER-1**	**WER-1**
Quran	Seq. Labeling	15.1	22.0	7.6	13.5
	Madamira	14.5	26.4	10.2	15.6
	Farasa	**7.8**	**14.0**	**5.0**	**6.8**
RDI	Seq. Labeling	16.7	28.0	13.6	12.0
	Madamira	12.5	20.4	8.6	10.2
	Farasa	**8.3**	**13.8**	**5.0**	**5.1**
Tashkeela	Seq. Labeling	24.0	35.4	15.0	22.0
	Madamira	12.4	20.3	8.5	10.1
	Farasa	**8.3**	**13.9**	**5.0**	**5.1**

Table 5: Error rates in relaxed evaluation mode.

Mona Diab, Mahmoud Ghoneim, and Nizar Habash. 2007. Arabic diacritization in the context of statistical machine translation. In *Proceedings of MT-Summit*.

Nizar Habash. 2010. Introduction to Arabic natural language processing. *Synthesis Lectures on Human Language Technologies*, 3(1):1–187.

Nizar Habash, Owen Rambow, and Ryan Roth. 2009. MADA + TOKAN: A toolkit for Arabic tokenization, diacritization, morphological disambiguation, POS tagging, stemming and lemmatization. In *Proceedings of the 2nd international conference on Arabic language resources and tools (MEDAR), Cairo, Egypt*, pages 102–109.

Osama Hamed and Torsten Zesch. Generating Nonwords for Vocabulary Proficiency Testing.

Osama Hamed and Torsten Zesch. 2017. The Role of Diacritics in Designing Lexical Recognition Tests for Arabic. In *3rd International Conference on Arabic Computational Linguistics (ACLing 2017)*, Dubai, UAE. Elsevier.

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

Yasser Hifny. 2012. Smoothing techniques for arabic diacritics restoration. In *12th Conference on Language Engineering*, pages 6–12.

John Lafferty, Andrew McCallum, and Fernando Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data.

Kristin Lemhöfer and Mirjam Broersma. 2012. Introducing LexTALE: A quick and valid lexical test for advanced learners of English. *Behavior Research Methods*, 44(2):325–343.

Aya S Metwally, Mohsen A Rashwan, and Amir F Atiya. 2016. A multi-layered approach for arabic text diacritization. In *Cloud Computing and Big Data Analysis (ICCCBDA), 2016 IEEE International Conference on*, pages 389–393. IEEE.

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 *LREC*, pages 1094–1101.

Mohsen Rashwan, Mohamed Al-Badrashiny, Mohamed Attia, Sherif Abdou, and Ahmed Rafea. 2011. A stochastic arabic diacritizer based on a hybrid of factorized and unfactorized textual features. *Audio, Speech, and Language Processing, IEEE Transactions on*, 19(1):166–175.

John Read. 2004. Plumbing the depths: How should the construct of vocabulary knowledge be defined. *Vocabulary in a second language*, pages 209–227.

Robert Ricks. 2015. The Development of Frequency-Based Assessments of Vocabulary Breadth and Depth for L2 Arabic.

Norbert Schmitt. 2014. Size and depth of vocabulary knowledge: What the research shows. *Language Learning*, 64(4):913–951.

Tzu-Hua Wang. 2007. What strategies are effective for formative assessment in an e-learning environment? *Journal of Computer Assisted Learning*, 23(3):171–186.

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