Deanship of Graduate Studies

Al-Quds University

Towards the Automatic Generation of Arabic Lexical Recognition Tests

Mohammad Khalil Ahmad Nassar

M.Sc. Thesis

Jerusalem – Palestine

1441/2020

Towards the Automatic Generation of Arabic Lexical Recognition Tests

Prepared by:

Mohammad Khalil Ahmad Nassar

B.Sc.: Computer Information Technology - Arab American University - Palestine

Supervisor: Dr. Saeed Salah

Co-Supervisor: Dr. Raid Zaghal

A Thesis Submitted in Partial Fulfillment of Requirements for the Degree of Master in Computer Science / Department of Computer Science / Faculty of Graduate Students / Deanship of Graduate Studies / Al-Quds University.

1441/2020

Deanship of Graduate Studies

Al-Quds University

Computer Science

Thesis Approval

Towards the Automatic Generation of Arabic Lexical Recognition Tests

Prepared by: Mohammad Khalil Ahmad Nassar

Registration No.: 21511730

Supervisor: Dr. Saeed Salah

Co-Supervisor: Dr. Raid Zaghal

Master thesis submitted and accepted, Date: 29/08/2020

1- Head of Committee: Dr. Saeed Salah	Signature ...[illegible]...
2- Co-Supervisor: Dr. Raid Zaghal	Signature ...[illegible]...
3- Internal Examiner: Dr. Radwan Kasrawi	Signature ...[illegible]...
4- External Examiner: Dr. Nael Abu Halawa	Signature ...[illegible]...

Jerusalem – Palestine

1441 / 2020

Dedication

To all who struggle for justice and create a better life for all Palestinians through adopting and having powerful knowledge that supposed to support our nation and push us staying on our holy land.

Mohammad Khalil Ahmad Nassar

Declaration

I certify that this thesis submitted for the degree of Master in Computer Science is the result of my own research, except where otherwise acknowledged, and that this thesis (or any part of the same) has not been submitted for a higher degree to any other university or institution.

Signed: [illegible]

Mohammad Khalil Ahmad Nassar

Date: 29/08/2020

Acknowledgements

I would like to express gratitude to my supervisors Dr. Saeed Salah and Dr. Raid Zaghal who inspired and taught me the patience to complete this thesis based on proper scientific research methods.

Also, I would like to thank the Master committee of the Department of Computer Science who supported me and treated me with all kindness and friendliness.

Thank you.

Mohammad Khalil Ahmad Nassar

Abstract

Lexical Recognition Test (LRT) themes are one of the main methods that are widely used to measure language proficiency of some common languages such as German, English and Spanish. However, similar researches for Arabic language are at development stages, and existing proposals mainly use human-generated methods. In this thesis work, we suggested a new methodology, based on a newly developed algorithm that aimed to design and construct an Arabic LRT. The developed algorithm generates nonwords dynamically based on Arabic language special characteristics. The main four characteristics that this developed algorithm considers are: orthography (spelling), phonology (pronunciation), n-grams and the word frequency map, which is an important factor to create a multi-level test. The developed algorithm differs from previous approaches in the sense that the previous approaches used Markov models to create nonwords while the developed algorithm use some of Arabic language letter characteristics to create high quality nonwords.

With the help of a large processed dataset of vocabularies (14,000,849), the developed algorithm was experimented. For this purpose, a Web-based application, following the suggested methodology, was designed and implemented to facilitate the process of setting up the LRT, and to manage and analyze learners' responses. The experimental results have shown that the LRT questions that were automatically generated by the proposed system had confused the learners, this is clear from the output of the confusion matrix which showed that 1/3 of the generated nonwords were able to distract the learners. Each vocabulary item had (49) responses; responses for real words (48% correct answers, 52% in-correct answers). For the nonwords responses about 30% incorrect answers; this means the system was able to confuse the learner by selecting them as real words, and 70% correct answers; this means that the responses did not confuse the learner. Consequentially, the results of recall and precision have smaller values, 0.28 and 0.54, respectively.

The study also analyzed other study dimensions towards achieving test scores. These dimensions are word length, word type, and knowledge of Arabic as the number of learning years, learner's main language, and gender. The results have shown that the most affecting dimension was the type of generating the nonwords, especially the orthographical one, and it would be better when the replacement letter is located in the intersection of both orthographical and phonological similarity groups, since most of the confusing vocabularies (277) were belonged to this deterministic item.

To validate the accuracy of the developed approach, we developed a version of the Arabic LRT. This version consisted of two sections: real words and nonwords. The nonwords section had been divided into two equal parts; vocabularies that were automatically generated from the developed algorithm, and the second part contained vocabularies that were generated manually by Arabic language expert, who used the same rules being implemented in the algorithm. The comparative study showed that results the accuracy of both methods is almost the same.

Table of Contents

List of Tables

List of Figures

Chapter 1: Introduction

1.1 Background

Natural Language Processing (NLP) is a set of techniques that interact between Artificial Intelligence (AI) and linguistics [1]. These techniques can be used for several purposes such as machine translation [2], text mining and processing [3], spelling auto corrections [4], Optical Character Recognition (OCR) applications, sentimental analysis, generation of automated languages placement tests, exam auto-corrections, voice recognition and others [5-6].

There are many research efforts that tackled Arabic NLP (ANLP), both for written and spoken parts [7]. However, ANLP is still facing many problems that emerged from Arabic characteristics [8]. Among these problems are the high diversity of the real spoken language “the dialect” [9] (Moroccan, Egyptian, Levantine, Gulf and Mesopotamian). Also, each general dialect, *e.g.*, Levantine has its own branches (Lebanese, Jordanian, Syrian and Palestinian), and each branch has its own dialect. For example, the Palestinian dialect includes some discrepancies between different cities, and even the villages as (Jerusalem, Hebron, Nazareth, Nablus, Salfit and Jenin) [10]. Local people can easily identify each dialect of these different Palestinian regions; this high diversity creates an Arabic “Diglossia”, which means that people use the different words to express the same object. To overcome this phenomena, researchers had raised a solution to collect each vocabulary from these dialects as it was recently done for Palestinian, Iraqi, Algerian [11], Saudi and Egyptian dialects that had been collected in one corpus [12, 21 - 22], but it is still not covering all Arabic dialects. The above issues have an impact when collecting datasets of Arabic language to be used in nonwords generation. Arabic language still does not have a complete collection that contains all possible items, this will affect the accuracy when classifying a given vocabulary as a nonword because it is not found in the dataset.

Another issue that faces ANLP is the Arabic script that indicates the style of writing Arabic letters with respect to letter position (first, middle and last) [13], many letters have different shapes when located in various positions [14]. In addition to that diacritization has a special meaning and position. These specificities make ANLP a complex issue [15], that are only stick to Arabic language among other languages. This behavior makes it difficult to include the diacritization in Arabic LRT because the high ambiguity that the diacritization might give to a certain word.

1.2 Problem Statement

Current problem could be noticed when learners cannot find an effective measure to indicate their levels of recognition and knowledge in Arabic language. The research problem was raised when trying to extend the current approach of existing LRTs that are used to generate high quality nonwords in Latin languages like English and German. Arabic language is a different case since it is right to left language, it has a diacritization "TASHKEEL" [16, 19], which is a semi-character associated with different letters of a given word, these diacritization could be located above, under or after the concrete letter, and in each situation the whole meaning of the given word will be different.

Moreover, Arabic language has five main dialects (Gulf, Mesopotamian, Levantine, Moroccan and Egyptian). Dialects create the phenomena of "Diglossia" which is having the same meaning with different words. On the other hand, Arabic language has 12 Million real words, some of them are being used, and this might induce misleading when someone tries to guess if a given vocabulary is a real word or a fake word. Furthermore, the scarce of researches in Arabic NLP domain and the lack of resources could be considered as another challenge. Despite the fact that there is a number of commercial corpuses that can be used to help in conducting a relevant research, considering more than one corpus from various resources enhance the process of classifying the vocabularies, and thus will reduce the error rate.

The above Arabic language specificities might affect the probability to create an Arabic language LRT version. Therefore, the developed approach tried to tackle these issues by developing a new algorithm that considered Arabic language characteristics to create good nonwords to be used in designing Arabic LRT version. By following this method, we avoided overhead that might be induced when trying to extend the algorithms being used in other languages.

1.3 Hypothesis

According to the literature review, as will be discussed later in Chapter 2, previous work showed that the LRT methods being applied for multiple Latin languages either use the manual generation of LRT main components (nonwords) or use semi-automated mechanisms. Consequently, we hypothesized that the LRT concepts could be adopted towards Arabic language to create Arabic LRT version when applying concepts of LRT and project them towards Arabic language. Therefore, Arabic nonwords could be generated by considering Arabic language characteristics (orthographic and phonological letters similarities group) and use them to generate fake words using an automated algorithm.

In the meantime, we hypothesized that automated nonword generation could be used to formulate Arabic language LRT version with the same behavior when using manual generated nonwords when considering same rules of nonwords creation (word frequency: number of occurrences of a given word as registered in the processed dataset, orthographic similarities letter set, phonological similarities set).

1.4 Research Motivation

Arabic language is considered as one of the top six languages that people communicate around the world. In the Arab world, there is more than 422 Million persons who use Arabic as mother language. In addition to that we can notice that the highest interest for understanding Arabic reviews over the Web and social media to understand Arabs' opinions towards several subjects. Furthermore, Muslims around the word need to know better about their religion. Thus, they must obtain some levels of Arabic. Last but not least, the enthusiasm to enrich Arabic language research using NLP technologies is highly demanded by the research community on these days taking into consideration that there is a scarce of effective measures that evaluate Arabic language proficiency levels and LRT Arabic researches are still at development stage, and existing proposals mainly use human-generated methods.

1.5 Research Objectives

This thesis work aims at helping foreign learners of Arabic language and the interested parties to estimate their Arabic language proficiency levels. A systematic quick test based on the automatic generation of the Arabic LRT had been developed [17]. Each LRT version has its own implicit algorithm that could transform a certain input (concrete vocabulary) to produce a nonword that could confuse the leaner. In our case we developed our own algorithm that creates and manipulates Arabic nonwords. It takes into consideration the high complexity of the Arabic language compared to other languages.

1.6 Research Questions

As noted above, the LRT methods for Arabic language are still at development stages, and existing proposals mainly use human-generated methods. Therefore, the main research questions that this thesis research work tried to answer are:

Is it possible to design an Arabic LRT considering the most important characteristics of Arabic language compared to other languages?

What are the main criteria that must be considered to automate the process of generating high quality nonwords to be used in LRTs for Arabic language?

1.7 List of Contributions

The main contribution of this thesis is to develop an Arabic LRT version in an automatic way to help Arabic language learners to test their language proficiency levels. Based on this, the main contributions of this thesis are:

- Proposing a new algorithm that considers some Arabic language characteristics to automatically generate high quality nonwords that increase the complexity/difficulty of Arabic LRTs.
- Developing a Web-based application based on the proposed algorithm to manage LRT setup and to collect and analyze learners' responses.
- Developing a validation criterion to evaluate the correctness of the proposed approach. The validation was mainly based on human-intervention, a version of the test was written by Arabic expert following the same rules, and the obtained results were compared, analyzed and discussed.

1.8 Research Limitations

Throughout the execution of this research, several limitations could be highlighted. One of the major issues is the lack of freely available resources of large-scale dataset, since most of corpora are limited and restricted to news and social agencies, some others are commercial. Having very huge dataset will enhance the output accuracy because in the verification process the developed algorithm is checking the generated nonwords against the real words dataset inventory, this indicates that including as much as of real words will increase the accuracy and vice versa. In addition to that, the lack of a similar work on Arabic language, it is true that the LRT is already applied to some Latin languages, but it was not applied on Arabic language, this contributed to exceed some extra efforts during the implementation of the different stages of this research as there is no similar previous works to build on.

Another limitation factor faced us while conducting the Arabic language LRT test is the target learners. It is true that the Arabic LRT was mainly developed for non-native speakers, but due to COVID-19 pandemic, almost all language educational centers were closed. Hence, it was extremely impossible to make the exam locally at one of the language centers. To solve this issue, we developed a Web application for conducting the test purposes that contained the test items to facilitate reachability, but we obtained participation of non-native less than expected so most the participated learners were native people.

1.9 Thesis Organization

Besides the introduction chapter (Chapter 1), this thesis is divided into four chapters. They are listed below with a brief description about the content of each one of them.

Chapter 2 (Background and Literature Review): This chapter discussed and analyzed the relevant research works carried out on LRTs and made a comparison among them and the work presented in this thesis.

Chapter 3 (Research Methodology): This chapter detailed the research methodology, starting from data collection and preprocessing steps towards discussing the developed algorithms to generate nonwords and the associated LRT test.

Chapter 4 (Experimental Results): This chapter discussed the experimental testbed, the experimental results with discussions and analysis, as well as the validation step that was carried out to verify the accuracy of the approach.

Chapter 5 (Conclusions and Future Works): This chapter concluded the main finings of this thesis work, and shaded lights to some future works for further improvements.

Chapter 2: Background and Literature Review

2.1 Background

Linguists had been the focus of several research efforts with the aim of finding the best way that could help language learners to know their proficiency levels. For example, English Lexical Project [18] contains international standard tests that had been created and became a standard measurement of learner' level for a specific language. Among these tests are TOEFEL (Test of English as a Foreign Language) and ILETS (The International English Language Testing System). These two tests are widely used to measure English language proficiency level for various categories of academic and business classifications. Another short quick test that had been used to give an indication towards English learner proficiency and other Latin languages is the LRT, many experiments and researches, coming from different research centers in Europe, had worked on this type of research to prove this concept using a real test implementation.

Since this thesis work focused on Arabic LRT, in the following we discuss the most relevant contributions, and shed the light towards their main drawbacks. Therefore, we avoided the potential problems related to some similar experiments [18, 24] carried out to design this test previously. Thus, this historical background associated with each entity related to creating Arabic LRT as the components that have an effect while generating good nonwords like diacritization role [16 - 17, 19 - 20] and its' benefits, and data preparation which consists of Modern Standard Arabic language (MSA). We excluded classical and dialect parts due to the lack of resources. Also, we found that Palestinian dialect had been collected but it is still not representative for all Levantine dialects [12, 21 - 23].

In our developed algorithm experiment, we focused on investigating the most important issues. We started by explaining methods being used to generate nonwords, and classified these methods based on tests' generation methods: manual or automatic. We also listed the main tools and applications that were implemented for this purpose, and mainly those that are used to prove the effectiveness of the process of generating the nonwords.

2.2 Arabic Language Challenges

In this section, we presented an overview towards Arabic Language distribution, categories, problems and processing challenges. Arabic is the six top used languages. It has three main categories: classical Arabic (CA), Modern Standard Arabic (MSA) and Arabic Dialect (AD). MSA and AD could be written either in Arabic or in Roman script (Arabizi), which corresponds to Arabic written with Latin letters, numerals and punctuation [25]. Due to the complexity of this language and the number of corresponding challenges while being processed when combining artificial intelligence and linguistics rules through implementing of Natural Language Processing (NLP). Many researches have been conducted, in order to facilitate analyzing Arabic content. One of the major issues that produced these problems are the diversity of morphology (wring script discrepancies, one word could be written using combination of different letters). These challenges were resolved by conducting huge work that intended to collect Arabic data from the internet and available books, then perform manual data cleaning. The obtained data was stored in a corpus files to be used as reference of scientific research. Such research is directly related to our research as it is bridging the gap of data lack, improving NLP methodologies to process Arabic language and decease the ambiguity of Arabic vocabularies by having reviewed version [26].

2.3 Manual LRT – Lextale Project

Lextale is a measurement for language proficiency applied for English, Dutch and German languages [24]; Lextale is a five minute (YES, NO) vocabulary identification test; it shows good results when indicating a vocabulary dataset, but it is still substantial when comparing it with other language proficiency tests like TOEIC (Test of English for International Communication), where users could apply this test through accessing this Web site *https:// www.lextale.com*. Lextale test consists of 60 (YES, No) questions, 40 words and 20 nonwords. Nonwords were generated and created manually, but the process of generating these nonwords should be efficient, and the generated nonwords must look like real words that could distract foreign learners from identifying them easily.

Lextale is considered as a good measurement for nonnative English language speakers having levels from medium to high. Lextale for Dutch and German are still not classified as a good measure. The manual generation of Lextale tests is also available. This manual process creates nonwords by replacing certain characters within the target word to obtain a similar nonword in terms of orthographic, phonological and morphological.

Validation to the generated Lextale tests was done by correlating its' results with other proficiency measurement tests such as Quick Placement Test (QPT). The test has been adapted to other languages beyond English, *e.g.,* Dutch and German, French [14] and Spanish [27].

2.4 English Lexicon Project

A manual generation of nonwords was adopted by English Lexicon Project5 (ELP) [18]. ELP is a large repository of databases (descriptive and behavior) linked to a search engine that aims to supply researchers with the necessary resources that could help them overcome the faced obstacles of processing the lexical tests. ELP could be accessed through the following Web site: *https://elexicon.wustl.edu.*

Data were collected from 1300 participants from six universities. Some of the exploratory information about this dataset is shown on the Website of the project that was mentioned above, it provides additional descriptive statistical data for the available words and nonwords and their frequencies. The ELP uses a manual procedure to create nonwords through replacing certain characters within the target word to obtain a nonword that is similar to the original one in terms of orthographic, phonological and morphological. Next, we explain these methods with the help of illustrating examples.

- Orthographic: This method uses the properties of neighborhood similarities to generate nonwords. As an example, consider the word "CAT", the list of orthographic words that are similar and returned from the ELP are the following.

  Neighbors of CAT: [OAT, COT, VAT, CAB, MAT, CAM, BAT, RAT, CAD, HAT, CAP, PAT, FAT, SAT, EAT, CAR, CUT, CAN].

- Phonological: This method uses the properties of neighborhood similarities of the character. Considering the same word "CAT".

  CAT /kat/ has three phonemes. *Number of syllables* provides the syllable count for a word. For example, CAT /kat/ has one syllable.

  The ELP returns the following phonological replacement based on the similarities of the pronunciation of the letter "C" with the pronunciation of the letter "K", so the syllable "kat" will replace the syllable "cat" where is it found, this intended to distract recognition of the generated non-word that having this syllable.

2.5 The ARC Nonword Database

In the ARC Nonword Database [19], the researchers provided a model based on phonological and orthographic rules that were applied to English of southern British. The results of this application are presented on the Website of this project with some statistical information. Items in this database were used to build the LRT test that is intended to strike the learner in different ways based on the morphological, orthographic and phonological rules.

2.6 Wuggy Research Project

Wuggy research project [28] developed a computer application that help researchers creating a better quality pseudoword or nonword following rules of languages, sub syllabic structure and transition frequencies between sub syllabic elements. It is already applied for multiple languages like Dutch, English, German, French, Spanish, Serbian, and Basque, and it could be expanded to other languages with some extra efforts. In this regard, pseudoword is considered as an important factor for lexical decision that represents a major tool used by psycholinguists to perform word processing tasks.

Because of the high effect of the nonwords on lexical decision performance, researchers invent an agreement towards the nonwords to be legal nonwords. This means that they must conform to the rules of generating real words in terms of orthographic and phonological.

Some of the limitations of the Wuggy algorithm are *(i)* it mainly depends on sub syllabic or summed bi-gram similarities; *(ii)* the program requires a user input called matching expression, so it is not fully automated solution for nonwords generation; *(iii)* the algorithm doesn't auto-detect the expression by which the word is ending.

Another similar application to Wuggy, called “WordGen”. WordGen is a tool for nonword selection and generation used in Dutch, English, German, and French. [27]. in this research both manual and automatic methods were used to generate nonwords.

2.7 Effect of Arabic Diacritization

Other researchers [12, 16 - 17] tried to show the important role of Arabic diacritized towards vocabulary assessment in the LRT, as they believed that diacritization reveals words ambiguity and makes better judgement while students identify the words. For this purpose, a sample for diacritized version of Arabic lexical test was generated along with a non-diacritized version to show the role of diacritization. The results have shown that the absence of diacritization increases the ambiguity of word’s identification. It’s worth mentioning that the most commonly written text in Arabic is a non-discretized, except in some historical, religious, and classical books, as well as in some specialized Arabic educational domains. Diacritization has an impact on nonwords design as Arabic diacritization is an orthographic way to describe Arabic word pronunciation [29].

2.8 Discussion

Hamed, O., *et al.* [17] suggested the use of a fully automatic methodology to generate high-quality nonwords that could be used in LRTs and confuse the learner. Discussing adaptation of automatic nonword generation paradigm is implemented on English language by Rastle, K., [19] to build a database of nonwords based on phonetical and orthographic properties of a given word. To apply the fully automated process of generating nonwords in English language, Hamed, O., [6] conducted a study of good nonwords generation using automatic methods by replacing a letter through an algorithm based on Markov Model and character language models, besides that the author had ranked the generated nonwords and used the highest ones in English LRT.

On the other hand, some researchers [17] investigated the Arabic LRT design and produced some studies that tackled finding role of Arabic diacritization (it is an orthographic way to describe Arabic word pronunciation, on concrete words and generated nonwords, in Arabic called "TASHKEEL"). This research [17] used a comparative study between nonwords of diacritized and non-diacritized and found that diacritization is empowering the nonword quality and it will be more robust to confuse the learner in the LRT test.

Compared to the previous research efforts, our developed methodology provided in this thesis work differs in the way how the approach being developed through the process of generating the nonwords of Arabic MSA (Arabic language letter characteristics).

This new approach produced main components of LRT tests which is high quality nonwords in Arabic that might be described as fake Arabic vocabulary that looks like a real word, and it was designed to distract the learner and confuse him/her in terms of pronunciation and writing shapes. These arguments were derived from Al-Ain book of the author al-Khalil ben Ahmad al-Farahidy [30] and it is mentioned above is our study contribution.

Chapter 3: Research Methodology

3.1 Research Method Overview

In this section, we overview the detailed stages of the research methodology. Figure 3.1 shows the main steps of the suggested methodology. As explained in the following sections. In the following subsections, we priovide more detaioled information about each step.

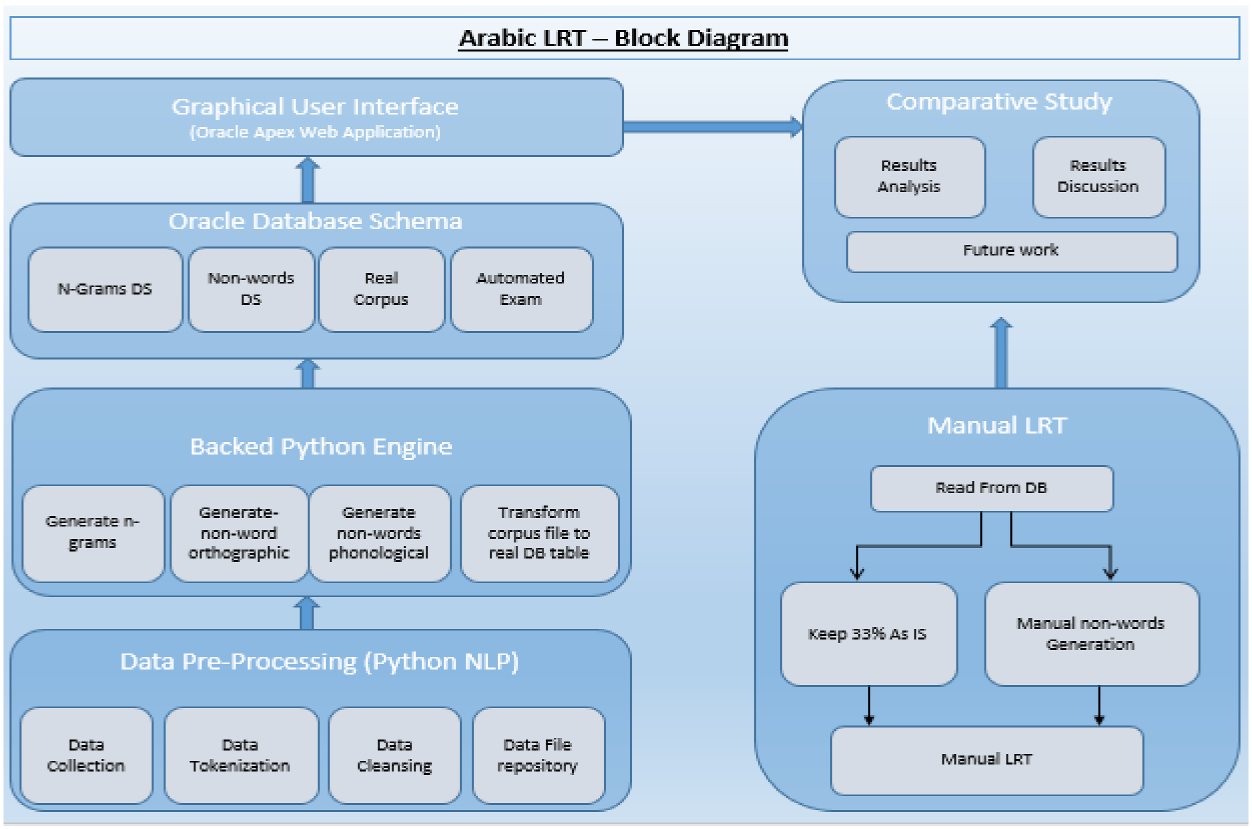


Figure 3. 1: Block Diagram of the Developed Approach of Arabic LRT Theme

3.2 Research Design and Method

The main methodology applied on this research considers auto-generating of Arabic nonwords based on replacing one letter in each word with a new letter which is similar to it in terms of the writing shape or phonetic. In addition to that, this method considers the frequency of a given word. This means that the higher word frequency, the higher familiarity of a word and vice versa, frequency has a value to tune difficulty level of delivered nonword, For more clarifications, main componenets and rules that formulate a high quality nonword in Arabic langauge are:

✓ Orthograpical Similarities

- Writing shape similarity between Arabic langauge alphabatical letters.
- Aims to switch a given letter with one of its orthographic similarity group to construct a nonwords that could confuse the learner.
- Table 3.1 shows Arabic lanagauge letters grouped into different orthographic categories.

✓ Phonological Similarities

- Phononitical pronunciation similarity (Place of articulation) between Arabic langauge alphabatical letters.
- Aims to switch a given letter with one of its phonological similarity group to construct a nonwords that could confuse the learner.
- Table 3.1 shows Arabic lanagauge letters grouped into different phonological categories.

✓ Word Frequency:

- It is the number of word occurrences.
- It is used to tune the algorithm to create multi-level LRT test.
- A word with high frequency will not confuse learners and vice versa.

✓ N-Gram:

- It divides the input word to its subsequent characters
- It is used to tune the algorithm for generating nonwords to get high quality nonwords.
- N-Gram example

['حا', 'ال', 'لي', 'يا', 'حال', 'الي', 'ليا', 'حالي', 'اليا', 'حاليا']

Table 3. 1: Summary of Arabic Language Letters’ Similarities Map

Similarity Type	**Similarity Set**
Orthographic	ح،ج،خ
Orthographic	ب،ت،ث
Orthographic	س،ص،ش
Orthographic	ذ،د
Orthographic	ض،ذ،ظ،ض
Orthographic	ق،ك،ف
Orthographic	ع،غ
Phonological-Place of articulation (velar-**الحلق**)	غ،ع،ح،خ، ه،ء
Phonological-Place of articulation (glottis-**اللسان**)	ت،ث،ج،د،ذ،ر،ز،س،ش،ص،ض،ط،ظ،ق،ك،ل،ث،ي
Phonological-Place of articulation (bilabial-**الجوف**)	ب،ف،م،و
Phonological-Place of articulation (oral cavity-**الشفتان**)	أ،و،ي

3.3 Data Collection and Preprocessing

Referring to the block diagram in Figure 3.1, in this thesis research study, a freely available corpora datasets had been collected from different resources as referenced for each one in the Table 3.2. These resources inherited from multi-category as news agencies, social media and Arabic books. For research consistency purposes, these dataset files were used in Arabic language projects. In the following subsections (data collection, dataset exploration), we summarized and visualized some information about these dataset (corpora).

3.3.1 Data Collection

Table 3.2 shows some technical information about the obtained datasets, each dataset has six dimensions represented as data fields as shown below, last data record in the table displays some aggregation summary information of the measurable dimensions:

Below is to highlight the dataset dimensions and references:

- ✓ Corpus source: the free source from which the data was obtained.
- ✓ File Name: each source could have one or more files.
- ✓ Char Count: number of characters.
- ✓ Lines: number of lines as existed in a notepad++ text file.
- ✓ Size: size in (KB) for each corpus.
- ✓ Diacritized or not
- ✓ Reference: corpus data source

All the listed datasets contained a huge number of raw data as assembled from the original resource as news agencies. Taking into considerations that there are other paid resources and as per our research purpose this free source is adequate to implement our methodology.

Table 3. 2: Summary of Raw Dataset (Dimensions and References)

Corpus Source	**File Name**	**Char Count**	**Lines**	**Size [KB]**	**Diacritized**	**Reference**
Al-Jazeera Corpus	aljazeera.txt	13,260,976	80,369	13,058	No	[31]
Al-Jazeera Corpus	aljazeera100 .txt	977,321	5,887	955	No	[31]
Books Corpus	books.txt	858,622	1,533	839	No	[32]
KACST Corpus	KACST.TX T	24,551,235	74,106	23,976	No	[33]
KACST Corpus	KACST100. txt	1,077,781	74,106	1,053	No	[33]
Al-Khaleej-2004 Corpus	khaleej.txt	27,283,987	5,695	26,645	No	[34]
Al-Khaleej-2004 Corpus	Khaleej100.t xt	1,106,419	231	1,081	No	[34]
Al-Watan-2004 Cor-pus	Wata100.txt	1,043,107	178	1,019	No	[37]
Al-Watan-2004 Cor-pus	Watan.txt	124,202,282	178	121,292	No	[37]
Watan Diac Corpus	Watan-diac.txt	163,473,924	40,579	159,643	Yes	[37]
Quran	quran.txt	743,918	6,236	727	No	[35]
RDI	rdi.txt	858,844	2,579	839	No	[38]
Tweets	Tweets-ann.txt	1,528,273	10,007	1,493	No	[39]
Tweets	Tweets-sharp.txt	1,514,713	10,007	1,480	No	[39]
Wackiness	WikiNew-sTruth.txt	177,279	423	174	No	[38]
Total		**362,658,681**	**312,114**	**354,274**		

3.3.2 Dataset Exploration

Referring to Table 3.2 and Figure 3.2, it is clearly observed that data files of the “Watan” news agency resource has occupied the most shares, while others have less data quota. On the other hand, Figures 3.2 and 3.3 show that the total number of non-diacritized words is higher than the number of the diacritized content. Meanwhile, this is not a big difference, but diacritized dataset has a high diversity since it is collected from multiple data sources coming from news agencies, social media, Quran and other books. While the diacritized content is derived from one data source.

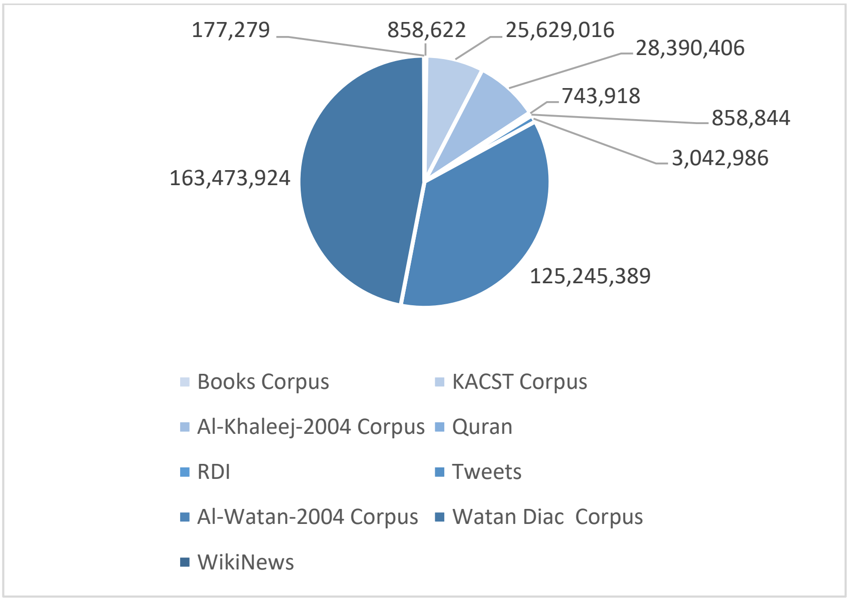


Figure 3. 2: Raw Dataset Visualization based on Source and Size.

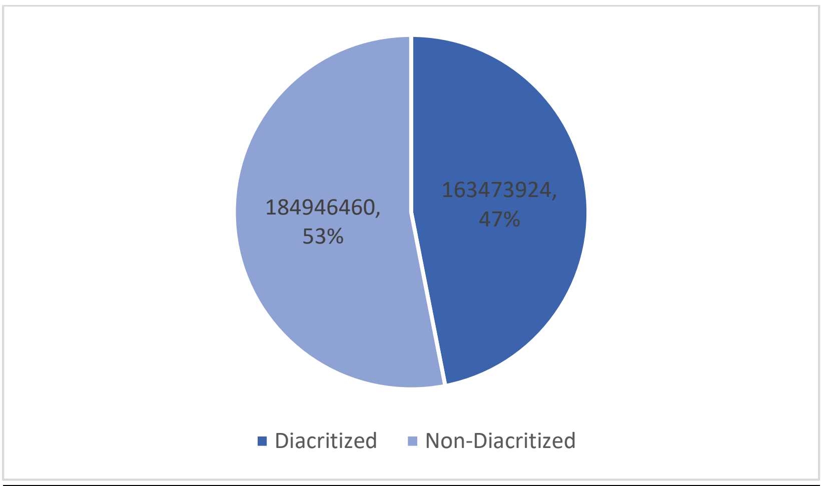


Figure 3. 3: Raw Dataset Visualization based on Diacritization and Size.

3.3.3 Data Tokenizer

Text tokenization is the process that take input parameter a stream of raw data and based on a delimiter as a white space or a specific character this data will be divided in smaller text based on the provided parameters [40]. In our research we required usage of text tokenization as an important step in the preprocessing of dataset stage.

Since the gathered data in Table 3.2 are in raw format, the tokenization process was applied to separate the content of each data file using a whitespace as a delimiter. This process is necessary to have each word in a separate line, and then to accumulate all results into one text file. In Figure 3.4, tokenization is represented by the steps of (Read Lines, Word Splitting), more technical detail about this process is available in the following python file Appendix 1.[Script Name: Dataset Prepare]

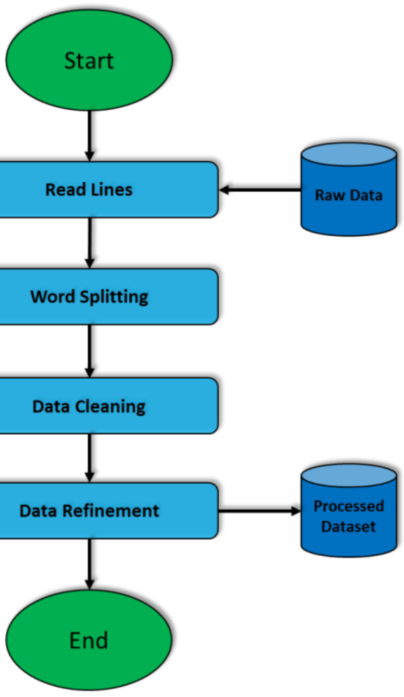


Figure 3. 4: The Flowchart of Data Preprocessing and Preparations Steps.

3.3.4 Data Cleaning

Data cleaning is the process of eliminating any undesired text content, In our case we only looking to keep the Arabic language words, so Figure 3.4 shows the data cleaning process that take input from data tokenization process, So during the cleaning process, the program eliminated Arabic nonword items including punctuation marks, special symbols, Arabic “TASHKEEL”, numeric values, stop words, one char length items, and any strange items. This was applied using python scripts as listed in Appendix 1. (Script Name: Dataset Prepare).

Table 3. 3: Summary of the Cleaned Dataset Words Resulted from Raw Dataset Preprocessing

Corpus Name	**Num. of Clean Words**
Al-Watan-2004 Corpus	85,052
Al-Jazeera Corpus	1,156,428
Al-Khaleej-2004 Corpus	2,272,750
Al-Watan-2004 Corpus	9,226,283
Books Corpus	74,770
KACST Corpus	2,036,728
Quran	66,314
RDI	74,959
Tweets	234,326

Table 3.3 contains the summary of the clean dataset words resulted from raw data preprocessing stage, now each corpus data is processed, and its content exists in a separate text file.

It is worth noting that we kept the duplicated information as received from each the source. We argue that data redundancy will have a significant value when generating nonwords, as it will be shown in the next chapter.

3.4 Nonwords Auto-Generation

Here, we are going to illustrate rules that used to create nonwords automatically through the developed algorithm, for a better understanding it is highly recommended to trace the pseudocode of the used algorithm as in Figure 3.4 and to go over Python source code that accomplished this task and available at Appendix 1. [Script Name: generatenonwords.py].

3.4.1 Nonwords Auto-Generation (Orthographic and Orthographic)

Creating nonwords in Arabic language was accomplished through using implementation of the pseudocode algorithm shown in Figure 3.5, the implementation already added to Appendix 1. [Script Name: generatenonwords.py] to illustrate this algorithm, below steps were added:

1. Word Frequency Calculation:

   The developed algorithm beings by looping through all cleaned vocabularies stored in the database. For each vocabulary, it calculates its frequency. To generate multi-level tests, the algorithm calculates the word frequency (Frequency); how many times the selected word appeared in the corpus.

2. Frequency Threshold:

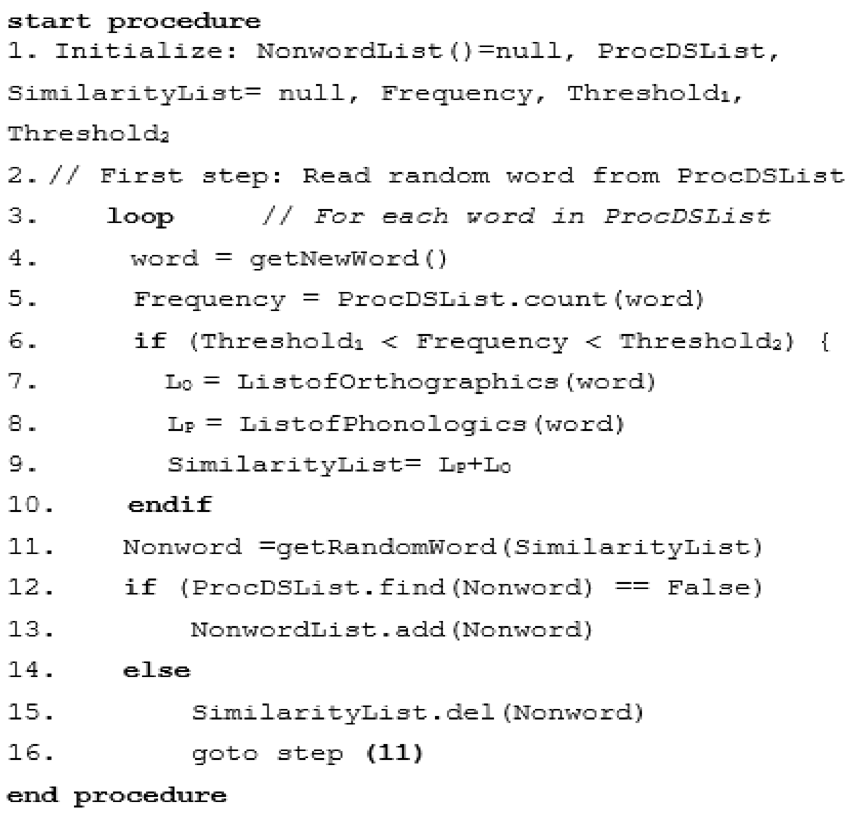
   Two thresholds were used (Threshold_1 and Thresholds_2) to tune the algorithm's operation. If $\text{Frequency} > \text{Threshold}_1$ && $\text{Frequency} < \text{Threshold}_2$ – the vocabulary is not used more frequently, the algorithm generates two lists: L_p; the list of orthographic vocabularies based on orthographic similarity map, and L_o; the list of phonological vocabularies based on phonological similarity map (Refer to Table 3.1). Next, it adds the two lists together to form a similarity list (SimilarityList), which contains all vocabularies generated from both orthographic and phonological similarity maps.

3. Nonwords Generation:

   To generate test's questions, the algorithm randomly selects a vocabulary from the SimilarityList, and checks the occurrence of this vocabulary in the processed dataset (ProcsDSList). If the conditional statements return FALSE - this means that the selected vocabulary is a nonword, it adds it to the (NonwordList) to be used by the LTR test. If the condition statement returns TRUE - this means that the selected vocabulary is considered as a real word, it removes it from the SimilarityList, and repeats the process again by selecting a new random vocabulary form the SimilarityList. For each generated nonword, the data record will store the ID of the original one, the replacement letter, the replacement position, and the new letter.

4. Python Script:

   Appendix 1. [File Name: generatenonwords.py] provides more technical details about how the algorithm generates the non-words.



```
start procedure
1. Initialize: NonwordList()=null, ProcDSList,
SimilarityList= null, Frequency, Threshold$_1$,
Threshold$_2$
2. // First step: Read random word from ProcDSList
3.    loop    // For each word in ProcDSList
4.      word = getNewWord()
5.      Frequency = ProcDSList.count(word)
6.      if (Threshold$_1$ < Frequency < Threshold$_2$) {
7.        $L_O$ = ListofOrthographics(word)
8.        $L_P$ = ListofPhonologics(word)
9.        SimilarityList= $L_P$+$L_O$
10.     endif
11.     Nonword =getRandomWord(SimilarityList)
12.     if (ProcDSList.find(Nonword) == False)
13.         NonwordList.add(Nonword)
14.     else
15.         SimilarityList.del(Nonword)
16.         goto step (11)
end procedure
```

Figure 3. 5: The Developed Algorithm for Nonwords Generation

3.4.2 Nonwords Sample (Automated Method)

Here, the automated process of generating the nonwords had considered word-frequency, word-length for both types; words and nonwords. Following these algorithmic rules, Table 3.4 shows the generated nonwords with the following details:

- AUTO_PHONOTICAL: It means that the replacement of character (CHAR_CH_ORG) that had been applied was based on Arabic phonological similarity rules.
- AUTO_ORTHOGRAPHIC*:* It means that the replacement of character (CHAR_CH_ORG) that had been applied was based on Arabic orthographic similarity rules.

Table 3. 4: Nonwords Sample – Auto-Generated Approach.

Real Word	Create Type	Word Length	Generated Word	Original Letter	New Letter
آيات	AUTOMATED_BOTH	4	آياث	ت	ث
ادعهن	AUTOMATED_BOTH	5	ادغهن	ع	غ
رتبتها	AUTO_ORTHOGRAPHIC	6	رتبثها	ت	ث
غيبتهم	AUTO_ORTHOGRAPHIC	6	غيببهم	ت	ب
موظفته	AUTO_ORTHOGRAPHIC	6	موظقته	ف	ق
تخذل	AUTO_PHONOTICAL	4	تخذق	ل	ق
سأنتهي	AUTO_PHONOTICAL	6	سأنتهل	ي	ل
صدقيني	AUTO_PHONOTICAL	6	صدكيني	ق	ك
مفاعل	AUTO_PHONOTICAL	5	مفاعط	ل	ط
ملامحا	AUTO_PHONOTICAL	6	ملومحا	م	و

- AUTOMATED_BOTH*:* It means that the replacement of character (CHAR_CH_ORG) that had been applied was based on both Arabic phonological and orthographic similarity rules. Below we provide more technical information about these generation methods.

3.4.3 Generating N-Grams

Character N-Grams are the subsequent characters of a word, this function loops through the cleaned data file, and then for each word, it generates all possible N-Grams starting form BI-GRAM to (word_Length - 1) GRAMS. These grams will be inserted into a database table with respect to the real word, this might be helpful to formulate a statistical data reference through which we can build some conclusions and judgements when tuning the nonwords generation. Since N-Gram could be involved in generating nonword by replacing a character in the input word taking into consideration frequency occurrence of prefix and postfix characters. Thus, n-grams are being used to narrow the acceptable possibilities; this is expected to increase the quality of the nonword generation process.

The implementation of this algorithm is explained in Appendix 1. [File Name: ngram-char.py]. The following is a sample of a new n-gram dataset that is generated when the word "**حاليا**" is being fed as an input to the ngram-char.py algorithm.

['حا', 'ال', 'لي', 'يا', 'حال', 'الي', 'ليا', 'حالي', 'اليا', 'حاليا']

3.5 Nonwords Manual Generation

3.5.1 Nonwords Sample (Manual Approach)

Following Arabic similarity rules, Table 3.5 shows the set of generated nonwords by Arabic expert following these rules:

- MANUAL_PHONOTICAL*:* It means that the manual replacement of character (CHAR_CH_ORG) that had been applied was based on Arabic phonological similarity rules.

- MANUAL _ORTHOGRAPHIC*:* It means that the manual replacement of character (CHAR_CH_ORG) that had been applied was based on Arabic orthographic similarity rules.
- MANUAL _BOTH: It means that the manual replacement of character (CHAR_CH_ORG) that had been applied was based on both Arabic phonological and orthographic similarity rules.

Table 3. 5: Nonwords Sample – Manual Generated Approach.

Real Word	**Create Type**	**Word Length**	**Generated Word**	**Original Letter**	**New Letter**
أبخرة	MANUAL_BOTH	5	أبجرة	خ	ج
أقترب	MANUAL_BOTH	5	أقثرب	ت	ث
أنحن	MANUAL_BOTH	4	أنخن	ح	خ
أتوقعها	MANUAL_ORTHOGRAPHIC	7	أتوكعها	ق	ك
فيت	MANUAL_ORTHOGRAPHIC	5	قيت	ف	ق
أتوق	MANUAL_PHONOTICAL	4	أتيق	و	ي
ذرع	MANUAL_PHONOTICAL	3	ذرغ	ع	غ
سعيد	MANUAL_PHONOTICAL	4	سغيد	ع	غ
طاغي	MANUAL_PHONOTICAL	4	طاغى	ي	ى
فأي	MANUAL_PHONOTICAL	3	فأو	ي	و

Chapter 4: Experimental Results

4.1 Processed Dataset Contents

Cleaned real words datasets produced in the data preprocessing stage and the nonwords produced in generation of nonwords stage are inserted into Oracle database schema tables when executing Python script illustrated in Appendix 1. [File Name: storeData.py], this step is performed to obtain on structured dataset instead of using text files data. This facilitate the upcoming processes of data analysis and manipulation besides configuring LRT templates, learners' dimensions and their responses. Finally, these data will be used in building the needed reports and dashboards, etc. Table 4.1 shows some statistical information about the aggregated data.

Table 4. 1: Summary of Real Words and Non-Words.

Item	Avg. Word Length	Count
Clean Dataset	6.5	14,000,849
Main Dataset-Distinct	6.5	399,495
Nonwords	5.2	38,412,714

The database schema tables are explained below:

- Real Words Dataset
    1. Contains all cleaned real data words.
    2. Appendix 2. [Table Name: lrt_corpus_filtered]

- Nonwords Dataset

1. Contains all auto-generated nonwords.
2. Appendix 2. [Table Name: lrt_nonwords]

- ➢ N-Gram Dataset

1. Contains all generated N-Gram words' segments.
2. Appendix 2. [Table Name: lrt_5_gram_summary]

- ➢ LRT Details Dataset

1. Appendix 2. [Table Name: lrt_exam]
2. Appendix 2. [Table Name: lrt_exam_details]

- ➢ Word Frequency Dataset

1. Appendix 2. [Table Name: lrt_word_frequency]

4.2 System Implementation

We will explore the project implementation steps and techniques that used to accomplish each stage. For the web application development, it was implemented through using Oracle APEX 19.1 framework. APEX is a rapid development framework from Oracle, and it is used to have a user-friendly interface through which learners can interact, register and take the test. The system administrator can use this interface to analyze test results and to create relevant reports and dashboards, here we summarized all these technologies and tools that had been used in designing and implementing Arabic language LRT version:

- ✓ Python v3.7 [Data Preprocessing Stage]
- ✓ Oracle Express Database v.12c [Data Manipulation, Data Storage]

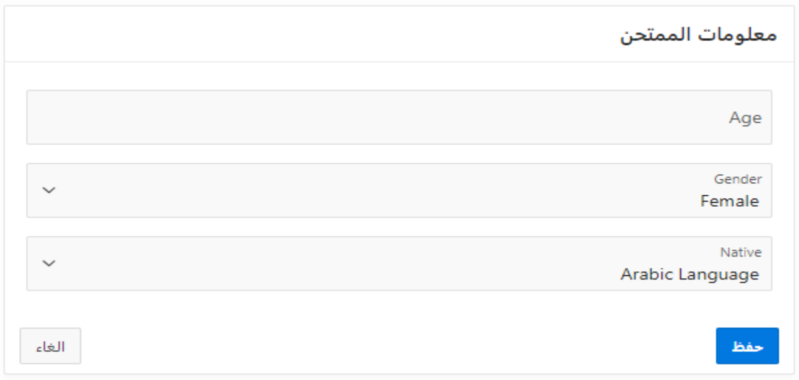
- ✓ Oracle Application Express APEX v.20 [Test setup, Exploratory Data Analysis]
- ✓ Google Forms [Test Setup]
- ✓ Microsoft Office [Analysis, Visualization]

Furthermore, we experimented the suggested algorithm using the processed dataset, the study was prepared through using a mix of manual and automatic LRT components in order to discover the effectiveness, validity and measurability of the developed approach. The target learners were both non-native and native with beginner to medium levels; Some dimensions of learners had been collected using Google forms instead of the developed web application because it was not hosted on a public domain so the Google form passed this task as it was prepared to facilitate and manage learners' responses and to analyze these results. At the beginning of the test session, the system requested from the learners to fill the following dimensions: age, gender, nativity, and number of years while learning Arabic; these details were stored in the database schema table, for more information refer to Appendix 2. [Table Name: lrt_exam]. Also, the impact of these dimensions on the obtained responses were analyzed in chapter 4 (Discussion and Conclusion).

It's worth mentioning that, this test was suggested to be taken by learners who registered in some well-known language centers, but due to COVID-19 pandemic and to facilitate reachability of students, a Google form was designed to be very similar to the developed Web application (local hosted, not on a public domain) and this form had been distributed among interested communities locally and globally.

4.2.1 Automated Generation of LRTs

An Oracle APEX workspace application had been created. This workplace had accomplished the task of creating the automated version of the LRT test. Oracle SQL statements were created to select and manipulate data of test tables in the database schema, some conditions were used in the query like frequency, vocabulary length and type (orthographic and phonological) to tune test difficulty and flavor. The relevant table structure is illustrated in Appendix 2. [Table Name: lrt_exam_details]. In the test window, some personal learner details are requested as native language, age, gender and number of years of learning Arabic. These dimensions data and the results of the given test will be used to analyze the test's dataset. For more details refer to Figures 4.1, 4.2, 3.3 and 3.4 that contain some snapshots of the learner dimensions and test's items in the LRT Web Application test.



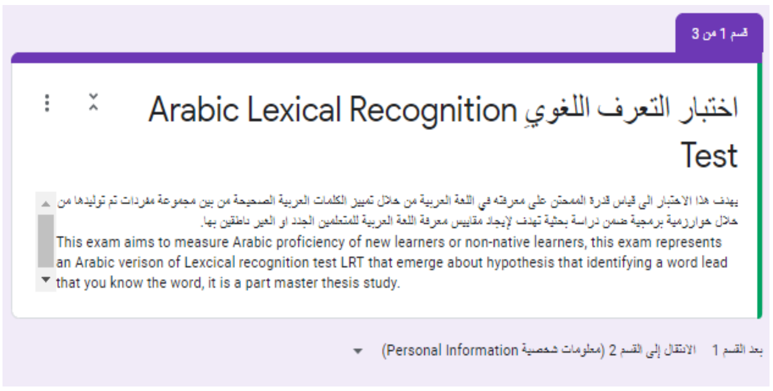
معلومات الممتحن

Age

Gender: Female

Native: Arabic Language

Figure 4. 1: A snapshot Web-application LRT Rest – learner dimensions.

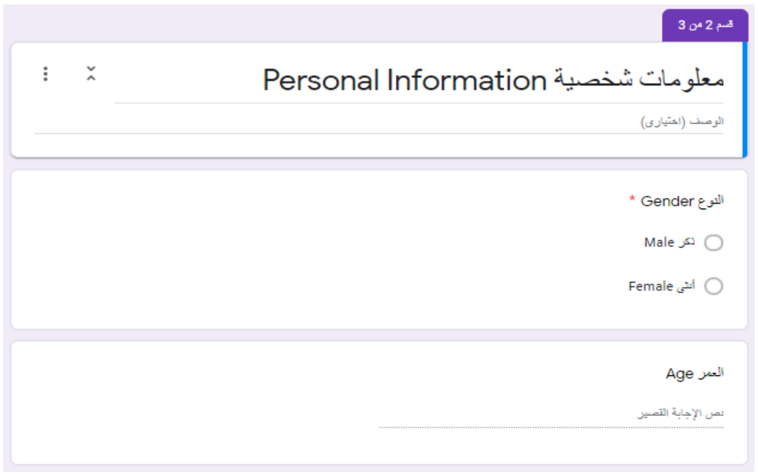

قسم 1 من 3

اختبار التعرف اللغوي Arabic Lexical Recognition Test

يهدف هذا الاختبار الى قياس قدرة الممتحن على معرفته في اللغة العربية من خلال تمييز الكلمات العربية الصحيحة من بين مجموعة مفردات تم توليدها من خلال خوارزمية برمجية ضمن دراسة بحثية تهدف لإيجاد مقاييس معرفة اللغة العربية للمتعلمين الجدد او الغير ناطقين بها.
This exam aims to measure Arabic proficiency of new learners or non-native learners, this exam represents an Arabic verison of Lexcical recognition test LRT that emerge about hypothesis that identifying a word lead that you know the word, it is a part master thesis study.

بعد القسم 1 الانتقال إلى القسم 2 (معلومات شخصية Personal Information)

Figure 4. 2: A Snapshot of LRT Google form Preface.


قسم 2 من 3

معلومات شخصية Personal Information

الوصف (اختياري)

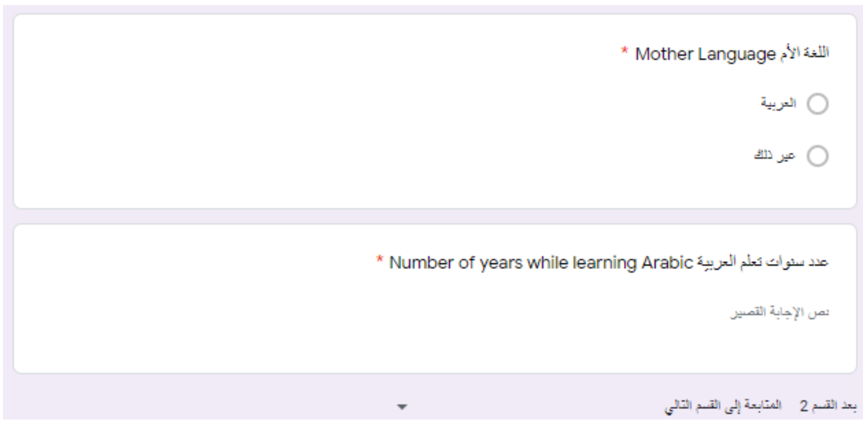
الوزن Gender *

- ○ ذكر Male
- ○ أنثى Female

العمر Age

نص الإجابة القصيرة

Figure 4. 3: A Snapshot of LRT Google form with Learner Dimensions - I.


اللغة الأم Mother Language *

- ○ العربية
- ○ غير ذلك

عدد سنوات تعلم العربية Number of years while learning Arabic *

نص الإجابة القصيرة

بعد القسم 2 المتابعة إلى القسم التالي

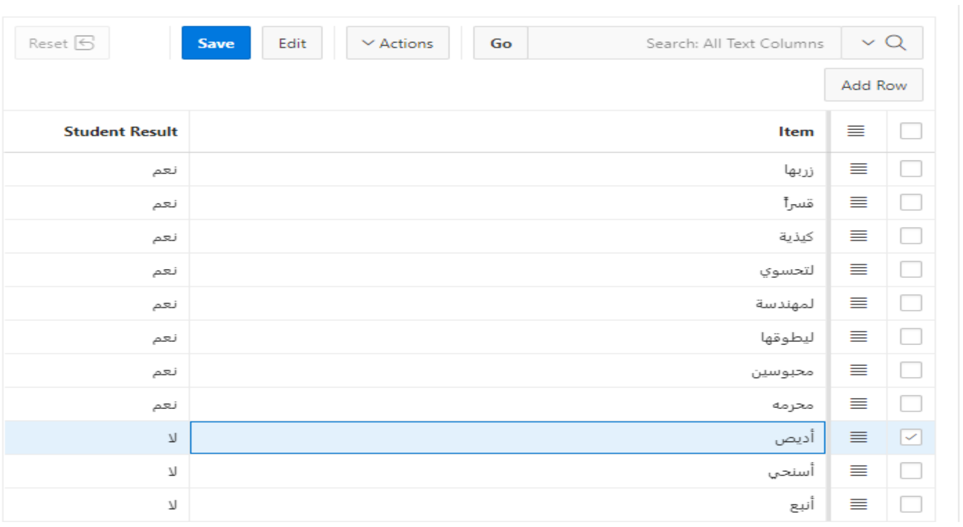
Figure 4. 4: A Snapshot of LRT Google Form with Learner Dimensions - II.

4.2.2 Manual Generation of LRTs

For purpose of results' validation, we conducted a comparative study by building a manual generation of nonwords (orthographic, phonological) to compare its responses with responses of the automated nonwords generated through the developed approach. To do that, we got help from an Arabic expert, who teaches Arabic language for many years. This expert had applied same rules of the automated approach. By applying this method, we were able to hold a comparative study between automatic generated nonwords and the manual generated nonwords.

4.2.3 Arabic LRT General Setup

Following to the previous versions of the LRT implemented in other languages, we prepared an LRT test with 1:2 (real words count: nonwords count). Thus, in this study, the created test contained 30 questions, 10 real words and 20 nonwords. For more information, refer to Figure 4.5. Note that the number of questions is 30, this number was chosen to be consistent with other placement tests that are mentioned in the literature review [17].



Student Result	Item		
نعم	زربها	≡	☐
نعم	قسرأ	≡	☐
نعم	كيذية	≡	☐
نعم	لتحسوي	≡	☐
نعم	لمهندسة	≡	☐
نعم	ليطوقها	≡	☐
نعم	محبوسين	≡	☐
نعم	محرمه	≡	☐
لا	أديص	≡	✓
لا	أسنحي	≡	☐
لا	أنبع	≡	☐

Figure 4. 5: A Snapshot that Shows the Automated Version of the Arabic LRT.

Since this is a comparative study that should contain manual and automated generation of nonwords, the 20 nonwords is divided into two groups (10 manually generated, 10 automatically generated) with possible answers (True, False) for each question.

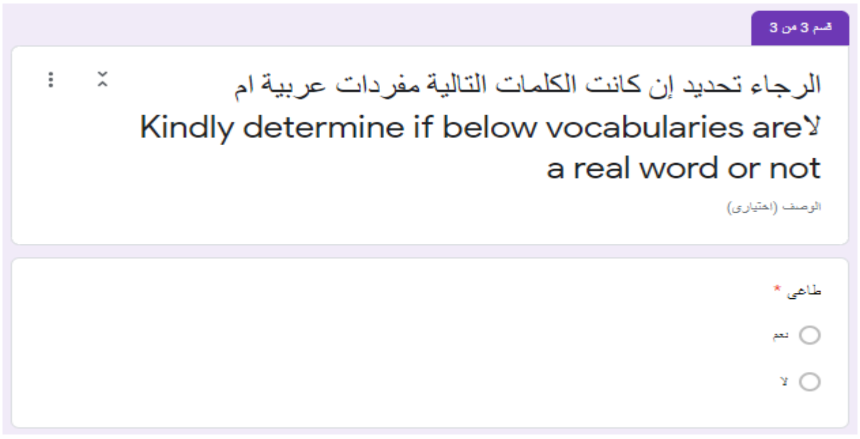
4.2.4 Arabic LRT Sample

Table 4.2 shows the real words sample that retrieved from the correct Arabic words' dataset stored in the table (Appendix 2. Table Name: lrt_corpus_filtered).

Table 4.2: LRT Sample Items and Details.

Real Word	Create Type	Word Length	Generated Word
أرنا	Real Word	4	أرنا
اجازوا	Real Word	6	اجازوا
تنويريا	Real Word	7	تنويريا
حجاجا	Real Word	5	حجاجا
عنبا	Real Word	4	عنبا
فضاءاته	Real Word	7	فضاءاته
فيعفيهم	Real Word	7	فيعفيهم
مقتوا	Real Word	5	مقتوا
يرسخوا	Real Word	6	يرسخوا
يهل	Real Word	3	يهل

The following snapshots (Figures 4.6, 4.7 and 4.8) were taken from the test version that was prepared on Google form.



قسم 3 من 3

الرجاء تحديد إن كانت الكلمات التالية مفردات عربية ام لا
Kindly determine if below vocabularies are a real word or not

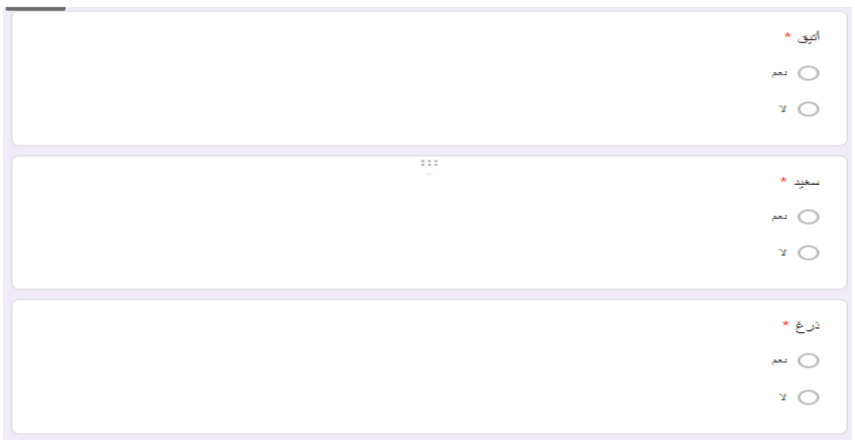
الوصف (اختيارى)

طاهى *

نعم

لا

Figure 4. 6: LRT Item in Google Form - I



أنيق *

نعم

لا

سعيد *

نعم

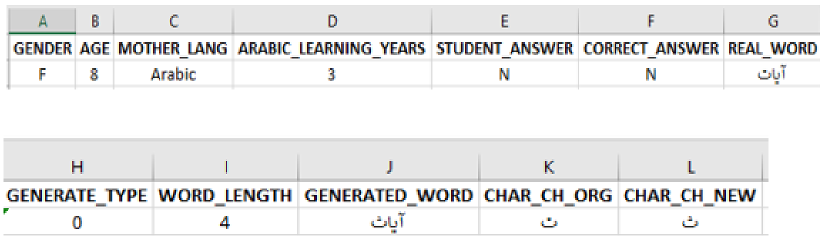
لا

درع *

نعم

لا

Figure 4. 7: LRT Item in Google Form – II



A	B	C	D	E	F	G
GENDER	AGE	MOTHER_LANG	ARABIC_LEARNING_YEARS	STUDENT_ANSWER	CORRECT_ANSWER	REAL_WORD
F	8	Arabic	3	N	N	آيات

H	I	J	K	L
GENERATE_TYPE	WORD_LENGTH	GENERATED_WORD	CHAR_CH_ORG	CHAR_CH_NEW
0	4	آياث	ت	ث

Figure 4. 8: Sample of LRT Learner’s Responses

For that, (49) learners’ responses for each test items were collected and saved into a csv file.

The header of the csv sheet is:

1. Gender: [F: Female, M: Male].
2. Age: Learner age.
3. Mother_Lang: [Arabic, Others].
4. Arabiclearning_Years: how long does the learner have been while learning Arabic?
5. Student_Answer: [Yes: if it is a real word, NO: if it is a nonword].
6. Correct_Answer: [Yes: if it is a real word, NO: if it is nonword].
7. Real_Word: The original word from which the nonword had been derived.

8. Word_Length: It gives the length of the real word; it is a deterministic dimension to be used in analyzing the results.
9. Generated_Word: It is a word that is generated when applying the modification rules.
10. Char_Ch_Org: It is the original char in the real words that had been replaced.
11. Char_Ch_New: It is a new substituted char that replaces the original one (CHAR_CH_ORG).

4.3 Exploratory Data Analysis

4.3.1 LRT Responses Analysis

Learners' data is collected through the used Google form as mentioned above, it contained the comprehensive (automatic, manual) test results, and these data were analyzed. We observed that the collected dataset has some redundancy. This form of redundancy has high impact on the nonwords generation process. Frequency has an inverse correlation with difficulty of the generated nonwords, and this conforms to the argument that the most common the word is, the easier to be known, and it is not easy to confuse the learner when replacing a letter with its similarity.

The second observation is that the average length of dataset words and nonwords is 6.5, and hence, the generated test has a query condition that determines this range of length to formulate the test items.

It is correct that the frequency had enhanced the flexibility to determine test's difficulty, and this supports the idea of multi-level test generation, but the drawback is having less distinct count when comparing it with the total Arabic real words which is about 12 Million words. This will affect the correctness ratio of the generated nonwords, while it is being classified as nonword.

Several generic evaluation measures were used to evaluate the performance of the developed approach. For the purpose of this work, we focused on the most common ones, specifically we consider accuracy, precision, recall, and F-score. The first three measures can be computed with the help of confusion matrix as shown in Figure 4.9. To ease the process of analyzing this figure, we provide the following definitions that are based on the work by Hamed, O., *et al.* [41].

1. True Positive (TP) is the number of correct answers, *i.e.,* positive class correctly identified as positive (real words that identified as real words).
2. True Negative (TN) is the number of correct answers, *i.e.*, negative class correctly identified as negative (Real words identified as nonwords)
3. False Positive (FP) is the number of incorrect answers, *i.e.,* negative class incorrectly identified as positive (nonwords identified as nonwords - learner was not distracted)
4. False Negative (FN) is the number of incorrect answers, *i.e.,* positive class incorrectly identified as negative (Nonwords that identified as real words – learner was distracted).

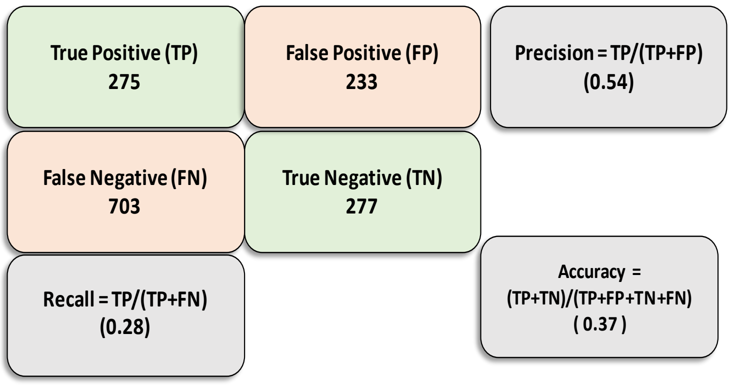


Figure 4. 9: The Confusion Matrix of the Learners' Responses.

4.3.2 LRT Responses Confusion Matrix

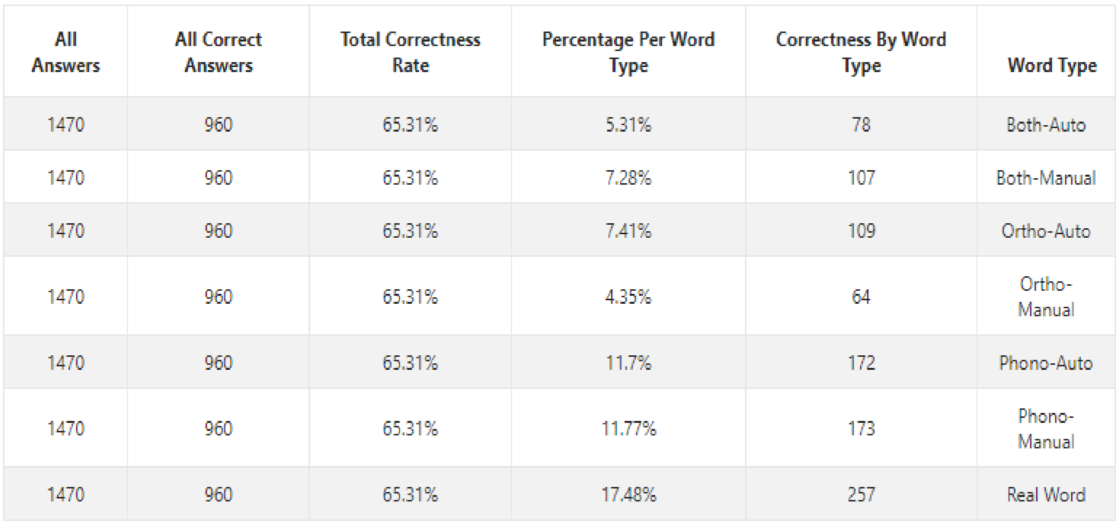
Figure 4.9 displays the confusion matrix of the learners' responses. The test consisted of (30) items, each one got (49) responses so the total is (1470) observation, (1/3) of total responses (1470) is coming from answers to the real words input set (10), so (470) observation are referring to answers of the real word set, among these (470) observations, (257) had correct answers but (233) had in-correct answers. From another side, observations of answers come from nonwords are (980), (277 out of 980) were able to distract the learner by considering them as real words, while (704 out of 980) were not able to distract the learner and they had given nonwords selection. From the output of the confusion matrix it is shown that (1/3) of the generated nonwords were able to distract the learners. Consequentially, the computed values of accuracy, recall and precision are (37%), (0.28 and 0.54), respectively. These small values indicate that the LRT questions that were automatically generated by the proposed system had confused the learners. It is correct that the frequency had enhanced the flexibility to determine test's difficulty, and this supports the idea of multi-level test generation, but the drawback is having less distinct count when comparing it with the total Arabic real words which is about 12 million words. This will affect the correctness ratio of the generated nonwords, while it is being classified as nonwords.

4.4 Results Discussions

Based on the above analysis, Table 4.3 summaries the main findings derived from applied test experiment. It is clearly shown that the observations fulfill expectations as we can find the main segregator dimension is the vocabulary generation type. In this study, robust nonwords that confused learners are the one which are generated based on the orthographic similarity rules, we want to make sure that the generated non-words are not real words.

In addition to this observation, the highest quality of nonword generation that achieved the least score is the nonword that has a replacement letter based on orthographic and phonological similarities. In other words, when the replacement letter could be in the intersection set between orthographic and phonological similarity groups.

Table 4.3: Summary of correct answers, correctness rate, percentage and correctness per word type.



All Answers	All Correct Answers	Total Correctness Rate	Percentage Per Word Type	Correctness By Word Type	Word Type
1470	960	65.31%	5.31%	78	Both-Auto
1470	960	65.31%	7.28%	107	Both-Manual
1470	960	65.31%	7.41%	109	Ortho-Auto
1470	960	65.31%	4.35%	64	Ortho-Manual
1470	960	65.31%	11.7%	172	Phono-Auto
1470	960	65.31%	11.77%	173	Phono-Manual
1470	960	65.31%	17.48%	257	Real Word

4.4.1 LRT Learner' Dimensions Impact

We explained the impact of different involved dimensions towards achieving the test's scores:

1. Word Length Impact:

Referring to Figure 4.10 that illustrates the relationship between the word length and the correct score, we found that learners were able to distinguish between real words and non-words when word length was in range of (4-6). For this case study, this means that this range was less confusing than being in length of 3 or 7 characters. This implies that generating non-words from a real word with length 3 or from a real word with length 7 or above, will produce high quality non-words and it is difficult to be identified when comparing them with non-words that had been generated from real words with length between (4-6), To generalize this finding more evidence is needed to reach such conclusions.

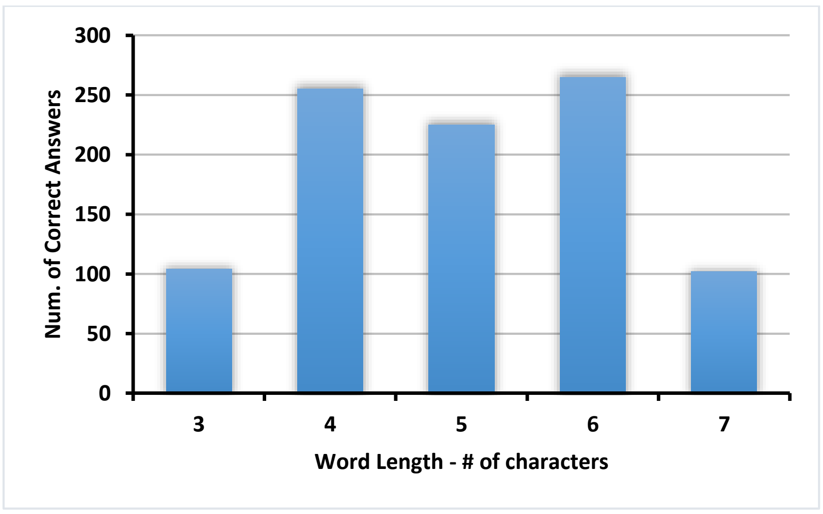


Figure 4. 10: Correlation between word length and the number of correct questions.

2. Word Type Impact:

By referring to Figure 4.11 that illustrates the relationship between word type and the number of correct answers, the highest learners' scores is achieved from the real words answers, we observed that most learners were native; on the other hand phonological got the second score, and this is expected since phonological replaces Arabic letters based on phonetical considerations, and this might make the generated nonword sounds strange. On the other hand, orthographic type had the minimum scores, and this is due to the argument that says if someone can identify the word then he/she knows it, and in the last cases learners had made identification because these nonwords were having high quality and really it was able to distract them.

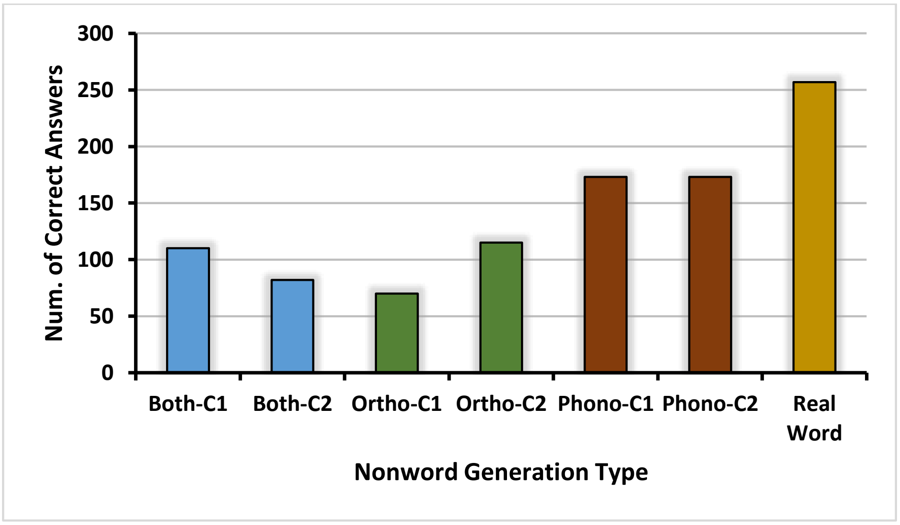


Figure 4. 11: A histogram that illustrates the relationship between word type and the number of correct answers.

3. Learning Years Impact:

Figure 4.12 depicts the relationship between the learner's language levels and the number of correct answers. It is concluded that the number of learning years could not be counted as a segregated item as the number of correct answers per learning years does not have a considerable variance among all learning levels.

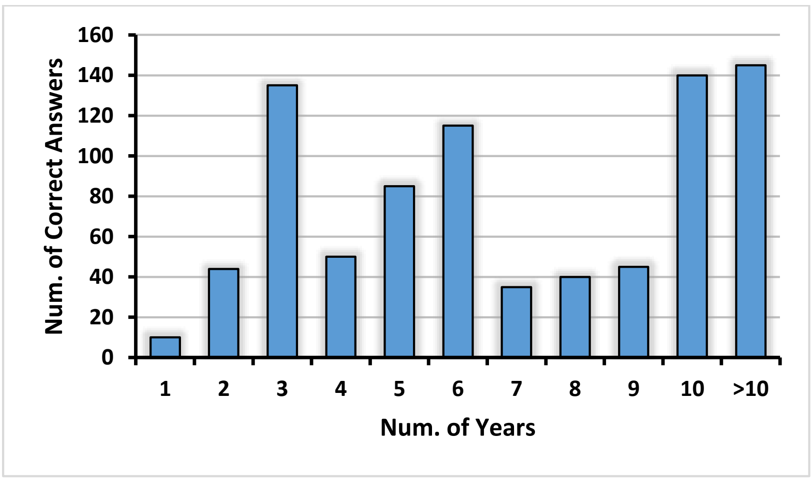


Figure 4. 12: A histogram that illustrates the relationship between the learner's language and correct answers.

4. Learners' Main Language Impact:

Figures 4.13 and 4.14 display the learners' responses distribution, where (1200 out of 1470) observations were produced by native learners, while (270) observations were produced by non-native learners. This high discrepancy explains why it is only one third of nonwords had confused the learners since native learners can identify their language better than non-native learners.

Figure 4.5 depicts that the total number of correct answers were (85%+15%) out of observations (270+1200) total operations. As shown in the figure, the most correct answers had been produced by native learners.

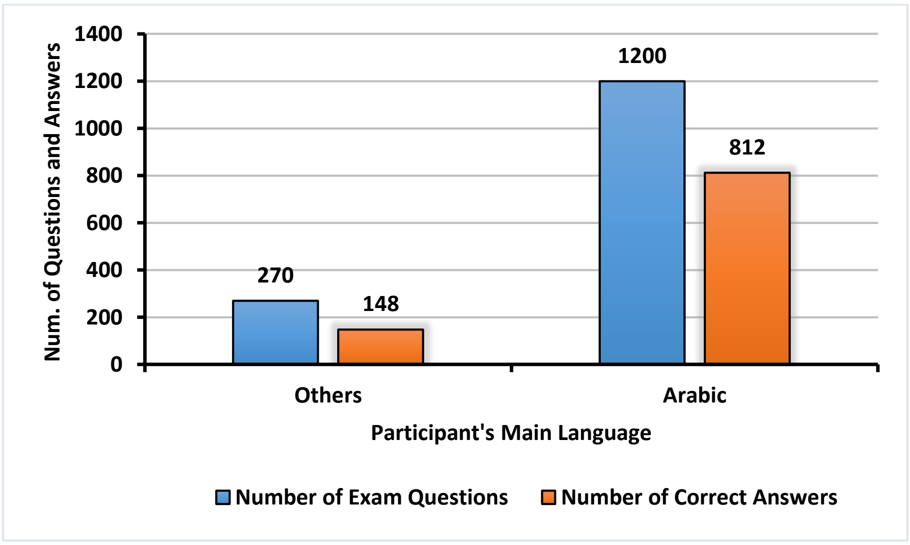


Figure 4. 13: Correlation between Main Language and Correct Answers.

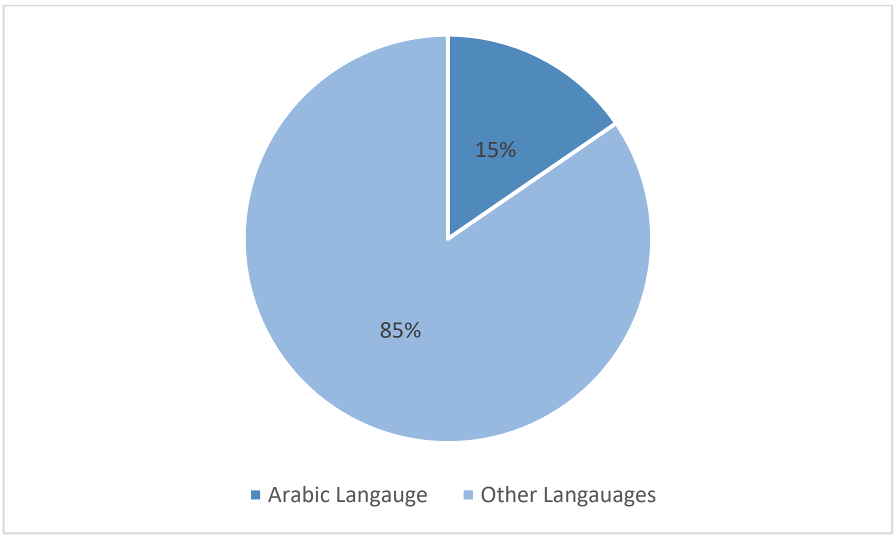


Figure 4. 14: Correlation between Participants' Main Language and the Correct Scores.

We could consider that mother language is a segregator, since most of the participated learners were Arab. To judge this conflict, we need to have similar groups to conduct comparative studies with closed values of suggested learner dimensions.

5. Gender Impact:

Figures 4.15 and 4.16 show the relationships between the numbers of correct answers vs. participant's gender. It is clear that most correct answers were achieved by males regardless their mother tongue, but in spite of this, we could not prove that males could have higher scores than females since male participant's count is much more than female count. To prove that gender is a segregation dimension, further investigation/research is needed to further explore this subject.

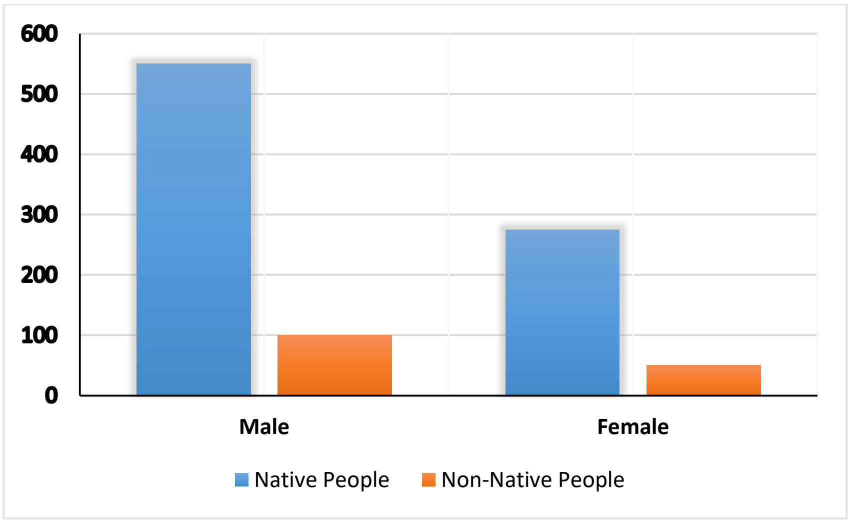


Figure 4. 15: Correlation between the Learners' Gender and the Correct Answers.

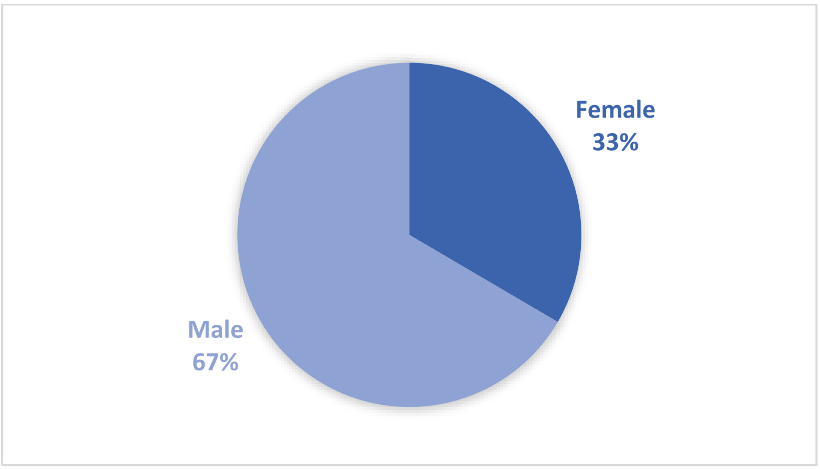


Figure 4. 16: Correlation between participant's main language and the correct scores.

4.4.2 Results Validation

A verification step was carried out to validate the above results. The main argument here is that generating good nonwords in Arabic is an eligible technique that can be used to establish Arabic LRT version. The results proved this argument as well. Besides this result, we could find that there is no contradiction between manual and automated methods, this indicates that automated nonwords generation is a valid option that could achieve results as the manual version prepared by a language expertise.

For obtaining more accuracy, more focus was given on generating nonwords based on letter replacement with its corresponding set, with the intersection between orthographic and phonological similarity groups. This intersection intended to produce high quality nonwords, taking into considerations words frequency rank.

Chapter 5: Conclusions and Future Works

5.1 Conclusions

Referring to results discussion we can find answers towards research questions, the main research question which related to check the ability to design high quality nonwords in Arabic language; we can observe that our empirical study developed a new algorithm that does this task with the help of Arabic character classifications theory (phonological and orthographic), as well as considering the word frequency map and n-grams concepts. The algorithm showed that generating high quality nonwords for Arabic language is a valid process, especially when the results validation achieved similar output when it is compared to other LRT versions. The output of the confusion matrix showed that one third of nonwords had confused learners, so the developed methodology based on a newly developed algorithm is capable to create high quality nonwords to be used in the Arabic LRT. This finding accomplishes one of the main objectives of the research which is creating good nonwords in Arabic language.

On the other hand, answering the second research question that inquiring the ability to design Arabic language LRT version; our empirical study showed that applying LRT concepts and methods on Arabic language is an adaptable process when applying Arabic language letters' characteristics (phonological, orthographic and word frequency) and it is approved when results of manual generated nonwords and auto-generated nonwords achieved similar results of learners' distraction and this finding achieving another objective of this research which is the adaptation and automation of LRT tests towards Arabic language.

Also, the developed solution presented a quick placement test that could be performed in (5-7) minutes, the results of the test gave a simple measurement indication of Arabic language knowledge; having such short evaluation will improve the process of classifying Arabic language learners' level. Having such empirical study will empower Arabic NLP researches which considered an achievement of another objective of this research, this addition is coming with a value of having real experiments and will be acting as a base of having much improvements and related works to build a multilevel evaluation platform of Arabic placement tests and so we can conclude that Arabic LRT research is a scalable and extendable research, this stage is being performed over vocabulary size; next time will could be done over sentence size or by considering word classes.

On the other hand, building a product based on this research became a real subject especially when considering main features of this created web application that can be summarized with high usability, reachability and the ability of creating supportive analysis and dashboards, this development will refine the process with multi improvements and ideas of utilizing Arabic properties in learners evaluation.

5.2 Future Works

As a future work, we are planning to include Arabic words diacritization "TASHKEEL", this is supposed to enhance Arabic language LRT version and make the generated nonwords more robust. Other future works for further improvements could be achieved by using classical Arabic language such as religious sources, traditional Arabic transcripts and literatures, this supposed to make the Arabic LRT version more comprehensive. Besides that, some further improvements on the test setup shall be considered such as grouping target learners and assign them to singular words type or plural or considering word categories (names, verbs, adverbs and plural), this classification intended to enrich Arabic language LRT’s as a specialized placement tests.

Since the developed method is capable to be extended to cover very large scale in terms of different learner levels by considering high diversity of Arabic language dialects [Levantine, Moroccan, Mesopotamian, Gulf and Egyptian]. Recently, some of listed dialects have been collected in one corpus. For example, Palestinian (one of Levantine dialects) and Egyptian dialects are available in one corpus for each. In addition to that, the generated tests with non-native students will enhance accuracy rate since native users find it easy to identify nonwords while the target audience of the research are non-native learners or beginner native learners. Finally, generated nonwords database could be used as an API service to check words spelling and to distinguish real words from incorrect words.

References

[1] Nadkarni, P. M., Ohno-Machado, L., & Chapman, W. W. (2011). Natural language processing: an introduction. Journal of the American Medical Informatics Association: JAMIA, 18(5), 544–551.

[2] Guzmán, F., Bouamor, H., Baly, R., & Habash, N. (2016, December). Machine translation evaluation for Arabic using morphologically-enriched embedding's. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers* (pp. 1398-1408).

[3] Badaro, G., Baly, R., Hajj, H., El-Hajj, W., Shaban, K. B., Habash, N., ... & Hamdi, A. (2019). A survey of opinion mining in Arabic: a comprehensive system perspective covering challenges and advances in tools, resources, models, applications, and visualizations. *ACM Transactions on Asian and Low-Resource Language Information Processing (TALLIP)*, *18*(3), 1-52.

[4] Shaheen, M., & Ezzeldin, A. M. (2014). Arabic question answering: Systems, resources, tools, and future trends. *Arabian Journal for Science and Engineering*, *39*(6), 4541-4564.

[5] Menacer, M., Mella, O., Fohr, D., Jouvet, D., Langlois, D., & Smaili, K. (2017, April). An enhanced automatic speech recognition system for Arabic. The third Arabic Natural Language Processing Workshop - EACL 2017, Apr 2017, Valencia, Spain.

[6] Abdul-Mageed, M. (2017, April). Not all segments are created equal: Syntactically motivated sentiment analysis in lexical space. In *Proceedings of the third Arabic natural language processing workshop* (pp. 147-156).

[7] Habash, N. Y. (2010). Introduction to Arabic natural language processing. *Synthesis Lectures on Human Language Technologies*, *3*(1), 1-187.

[8] Farghaly, A., & Shaalan, K. (2009). Arabic natural language processing: Challenges and solutions. *ACM Transactions on Asian Language Information Processing (TALIP)*, *8*(4), 1-22.

[9] Elfardy, H., & Diab, M. (2012). Aida: Automatic identification and glossing of dialectal arabic. In *Proceedings of the 16th eamt conference (project papers)* (pp. 83-83).

[10] Jarrar, M., Habash, N., Alrimawi, F., Akra, D., & Zalmout, N. (2017). Curras: an annotated corpus for the Palestinian Arabic dialect. *Language Resources and Evaluation*, *51*(3), 745-775.

[11] Bougrine, S., Chorana, A., Lakhdari, A., & Cherroun, H. (2017, April). Toward a Web-based Speech Corpus for Algerian Dialectal Arabic Varieties. In *Proceedings of the Third Arabic Natural Language Processing Workshop* (pp. 138-146).

[12] Jarrar, M., Habash, N., Alrimawi, F., Akra, D., & Zalmout, N. (2017). Curras: an annotated corpus for the Palestinian Arabic dialect. *Language Resources and Evaluation*, *51*(3), 745-775.

[13] Shahrour, A., Khalifa, S., Taji, D., & Habash, N. (2016, December). Camelparser: A system for arabic syntactic analysis and morphological disambiguation. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: System Demonstrations* (pp. 228-232).

[14] Zhang, L., Habash, N., & Toussaint, G. (2017). Robust Dictionary Lookup in Multiple Noisy Orthographies. In Proceedings of the Third Arabic Natural Language Processing Workshop (pp. 119-129).

[15] Hegazi, M. O. (2016). An Approach for Arabic Root Generating and Lexicon

Development. IJCSNS International Journal of Computer Science and Network Security, *16*(1).

[16] Hamed, O., & Zesch, T. (2017). The role of diacritics in designing lexical recognition tests for Arabic. *Procedia Computer Science*, *117*, 119-128.

[17] Hamed, O., & Zesch, T. (2015). Generating Nonwords for Vocabulary Proficiency Testing. In *Proceeding of the 7th Language and Technology Conference: Human Language Technologies as a Challenge for Computer Science and Linguistics* (pp. 473-477).

[18] Balota, D. A., Yap, M. J., Hutchison, K. A., Cortese, M. J., Kessler, B., Loftis, B. ... & Treiman, R. (2007). The English lexicon project. *Behavior research methods*, *39*(3), 445-459.

[19] Rastle, K., Harrington, J., & Coltheart, M. (2002). 358,534 nonwords: The ARC nonword database. *The Quarterly Journal of Experimental Psychology Section A*, *55*(4), 1339-1362.

[20] Al-Badrashiny, M., Hawwari, A., & Diab, M. (2017, April). A layered language model based hybrid approach to automatic full diacritization of Arabic. In *Proceedings of the Third Arabic Natural Language Processing Workshop* (pp. 177-184).

[21] Bougrine, S., Chorana, A., Lakhdari, A., & Cherroun, H. (2017, April). Toward a Web-based Speech Corpus for Algerian Dialectal Arabic Varieties. In *Proceedings of the Third Arabic Natural Language Processing Workshop* (pp. 138-146).

[22] Al-Twairesh, N., Al-Matham, R., Madi, N., Almugren, N., Al-Aljmi, A. H., Alshalan, S., ... & Al-Mutlaq, N. (2018). Suar: Towards building a corpus for the Saudi dialect. *Procedia computer science*, *142*, 72-82.

[23] Eskander, R., Habash, N., Rambow, O., & Pasha, A. (2016, December). Creating resources for dialectal arabic from a single annotation: A case study on egyptian and levantine. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers* (pp. 3455-3465).

[24] Lemhöfer, K., & Broersma, M. (2012). Introducing LexTALE: A quick and valid lexical test for advanced learners of English. *Behavior research methods*, *44*(2), 325-343.

[25] Guellil, I., Saâdane, H., Azouaou, F., Gueni, B., & Nouvel, D. (2019). Arabic natural language processing: An overview. *Journal of King Saud University-Computer and Information Sciences*.

[26] Salloum, W., & Habash, N. (2013, June). Dialectal arabic to english machine translation: Pivoting through modern standard arabic. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies* (pp. 348-358).

[27] Duyck, W., Desmet, T., Verbeke, L. P., & Brysbaert, M. (2004). WordGen: A tool for word selection and nonword generation in Dutch, English, German, and French. *Behavior Research Methods, Instruments, & Computers*, *36*(3), 488-499.

[28] Keuleers, E., & Brysbaert, M. (2010). Wuggy: A multilingual pseudoword generator. *Behavior research methods*, *42*(3), 627-633. *Ghent University, Ghent, Belgium.*

[29] Hamed, O., & Zesch, T. (2018, April). Exploring the effects of diacritization on Arabic frequency counts. In *2018 2nd International Conference on Natural Language and Speech Processing (ICNLSP)* (pp. 1-6). IEEE.

[30] Khalil, A., & Darwīsh, A. A. (1967). al-‘Ayn: First Arabic dictionary by al-Khalil ben Ahmad al-Farahidy. Baghdad: Maṭba‘at al-‘Ānī.

[31] “Al Jazeera Media Network” [Online; Last Accessed on: 28th August 2020]. Available: http://www.aljazeera.net/portal.

[32] “SourceForge Repository.” [Online; Last Accessed on: 28th August 2020]. Available: https://sourceforge.net/projects/tashkeela/.

[33] “SourceForge Repository.” [Online; Last Accessed on: 28th August 2020]. Available: https://sourceforge.net/projects/kacst-acptool/files/.

[34] “Google Sites” [Online; Last Accessed on: 28th August 2020]. Available: https://sites.google.com/site/mouradabbas9/corpora

[35] “Quranic project - Tanzil.” [Online; Last Accessed on: 28th August 2020]. Available: http://tanzil.net/download/

[36] “The Engineering Company for the Development of Digital Systems.” [Online; Last Accessed on: December 2017]. Available: http://www.rdi-eg.com/.

[37] “Association for Computational Linguistics **(**ACL**)”** [Online; Last Accessed on 28th August 2020]. Available: https://www.aclweb.org/anthology/D15-1299.

[38] “Association for Computational Linguistics **(**ACL**)”** [Online; Last Accessed on 28th August 2020]. Available: https://www.aclweb.org/anthology/W17-1302

[39] Nabil, M., Aly, M., & Atiya, A. (2015, September). Astd: Arabic sentiment tweets dataset. In *Proceedings of the 2015 conference on empirical methods in natural language processing* (EMNLP), Lisbon, Portugal, 2015, pages 2515–2519

[40] Zalmout, N., & Habash, N. (2017). Optimizing tokenization choice for machine translation across multiple target languages. In Proceedings of the European Association of Machine Translation (EAMT). *The Prague Bulletin of Mathematical Linguistics*, *108*(1), 257-269. Prague, Czech Republic, 2017

[41] Hamed, O., & Zesch, T. (2019). *Automatic Generation of Lexical Recognition Tests Using Natural Language Processing*. In: PhD Thesis, Faculty of Engineering, University of Duisburg-Essen, Duisburg, Germany (2019).

نهج التوليد التلقائي لإختبارت التعرف اللغوي في اللغة العربية

إعداد **:** محمد خليل أحمد نصار

المشرف الرئيس **:** د. سعيد صلاح

المشرف المشارك: د. رائد الزغل

الملخص

اختبارات التعرف اللغوي **(LRT)** مطبقة لدى لغات عالمية مثل الانجليزية، الالمانية والاسبانية. على اية حال هذه الاختبارات ما زالت في مرحلة التطوير في اللغة العربية، وان الاعمال البحثية المشابهة تستخدم النظام اليدوي لانساء مثل هذه اختبارات. في هذا العمل البحثي، أقترحنا طريقة جديدة مبنية على خوازمية جديدة تهدف لتصميم وبناء أشباه كلمات ذات نوعية عالية مرتبطة بمقياس سريع وحقيقي لتحديد مستويات إتقان العربية. الخوارزمية المقترحة تقوم بتوليد أشباه كلمات بالإعتماد على خصائص الكلمات والحروف العربية، الخصائص الاربع الرئيسية التي تعتمدها هذه الخوارزمية الجديدة هي: التشابه الكتابي، التشابه اللفظي في الكلمات، تقسيم مقاطع الكلمة، ودرجة تكرار الكلمات الاصلية. تعتمد هذه الخوازمية على فكرة تبديل حرف واحد بالكلمة المدخلة بحرف نظير له مأخوذ من مجموعة نظائر الحرف الاصلي الشكلية او اللفظية. تختلف الخوارزمية المقترحة عن الدراسات السابقة، حيث ان الدراسات السابقة استخدمت نماذج ماركوف لتوليد اشباه كلمات، بينما الخوارزمية المقترحة استخدمت بعض الخصائص المهمة في اللغة العربية.

بمساعدة قاعدة بيانات ضخمة تتكون من (14,000,849) كلمة، تم تجريب الخوارزمية المقترحة. للقيام بهذه المهمة، تمت تصميم وبرمجة واجهة انترنت تطبيقية لانشاء الاختبار، ولتسهيل ادارة اجبابات الممتحننين وتحليلها. نتائج التجربة اظهرت أن ثلث أشباه الكلمات المستخدمة في إختبار التعرف اللغوي قد شتت الممتحن. وهذه النتيجة واضحة من مخرج ال Confusion Matrix، التي اظهرت ان ثلث الكلمات التي تولدت بشكل اوتوماتيكي من الخوارزمية شتت اداء المتقدم للاختبار. تكونت عناصر الاختبار من 30 مفردة، عشرة منها كلمات حقيقية وعشرة اخرى هي أشباه كلمات قد

تم توليدها من خلال الخوارزمية وعشرة قد تم توليدها يدويا من قبل خبير لغوي، وكل واحدة من تلك المدخلات حصلت على 49 إجابة بمجموعع 1470 إجابة، و قد تم استنتاج 257 إجابة صحيحة للكمات الحقيقة و233 إجابة غير صحيحة، وكشفت الدراسة ايضا ان 277 أجابة خاطئة لأشباه الكلمات، أي ان ثلثها قد أربك الممتحن وكأنها كلمات حقيقية. نتيجة لذلك كانت قيم Precision Recall هي 0.28 و 0.54، بالترتيب.

و حللت الدراسة ايضا بعض العوامل ذات التأثير على قرار الممتحن وهي طول الكلمة، نوع الممتحن، عدد سنوات تعلم العربية ونوع المفردة في الإختبار، حيث كشفت النتائج أن نوع المفردة هو العامل الوحيد المؤثر، خاصة أشباه الكلمات الشكلية وأكثر منها أشباه الكلمات التي تحتوي على حرف بديل موجود في كلا المجموعتين الشكلية واللفظية، وأن معظم النتائج التي شككت الممتحن البالغة 277 إجابة هي من هذا النوع.

لمعرفة صلاحية و دقة الطريقة المقترحة، تم إستخدام إستبانات جوجل في بناء إختبار تعرف لغوي بنموذج عربي، إحتوى الإختبار على كلمات حقيقية واخرى أشباه كلمات قد تم توليدها من خلال الخوارزمية ومثلها تولدت يدويا، واظهرت النتائج ان نتائج الدقة في كلا الحالتين متشابهة.

Appendix 1. Application Code and Scripts

Script Name: Dataset Prepare (Raw Dataset Preprocessing)	
Process Name	Data Prepare: Data Tokenization
Description	Read raw data from corpora files, tokenize data, store data in one clean text file
Code File	DSPrepare.py
Dependency	Execution sequence: 1 This is the first python file to be executed once dataset is available in one text file
Python Code	from string import punctuation
	import re
	import pathlib
	class DSPrepare:
	arabic_punctuations = '''`÷×؛<>_()*&^%][ـ،/:"؟.,'{}~¦+|!”…“–ـ'''
	ar = ['د', 'ج', 'ح', 'خ', 'ه', 'ع', 'غ', 'ف', 'ق', 'ث', 'ص', 'ض',
	'ذ', 'ش', 'س', 'ي', 'ب', 'ل', 'ا', 'أ', 'إ', 'ت', 'ن','م',
	'ك', 'ط', 'ظ', 'ز', 'و', 'ة', 'ى', 'آ', 'لا', 'ر', 'ؤ','ء'
	]
	ar_tashkeel = ['ِ', 'ْ', 'َ', 'ّ', 'ُ', 'ٌ', 'ٍ', 'ً', 'ٓ', 'ً']
	def readFile(self):
	corpusPath="C:\\Users\\mohammad.nassar\\PycharmPro-jects\\MSProject\\MSThesis\\CorporaDS"

	ProcessedDSPath="C:\\Users\\mohammad.nassar\\PycharmPro-jects\\MSProject\\MSThesis\\ProcessedDS"
	WordStatisticeFile = open(ProcessedDSPath + "\\" + "WordStatis-ticeFile.txt" , 'a', encoding='utf-8')
	FileAppender_All_Tokenized = open(ProcessedDSPath + "\\" + "Tokenized_All_v5.txt", 'a', encoding='utf-8')
	TotalWordsCounter = 0
	for corpus_file in pathlib.Path(corpusPath).iterdir():
	fileWordsCounter = 0
	if corpus_file.is_file():
	if corpus_file.name.strip().endswith(".txt"):
	FileAppender1 = open(ProcessedDSPath+"\\"+ "To-kenized_v5_"+corpus_file.name, "a")
	with open(corpus_file, 'r+',newline='', encoding='utf-8') as f1:
	data = f1.readlines()
	for line in data:
	words = line.split()
	for word in words:
	if (word not in punctuation) and (word != '،') and (not word.isnumeric()) and not bool(
	re.search(r'\d', word)) and len(word) > 2:
	print(word)
	FileAppender1.write(word.strip() + "\n")
	FileAppender_All_Tokenized.write(word.strip() + "\n")

	fileWordsCounter = fileWordsCounter + 1
	WordStatisticeFile.write("Total Words in file "+ "Tokenized_v5_"+ corpus_file.name + " = ")
	WordStatisticeFile.write(str(fileWordsCounter))
	WordStatisticeFile.write("\n")
	TotalWordsCounter=TotalWordsCounter+fileWordsCounter
	f1.close()
	FileAppender1.close()
	WordStatisticeFile.write("\n\n")
	WordStatisticeFile.write("Total Words in all file = ")
	WordStatisticeFile.write(str(TotalWordsCounter))
	WordStatisticeFile.close()
	FileAppender_All_Tokenized.close()
	r1 = DSPrepare()
	r1.readFile()
	The End

Script Name: Nonwords Generation Using the Developed Algorithm (generatenonwords.py).	
Process Name	Nonwords generations
Description	Nonwords generations based on orthographic, phonological Arabic language rules.

Code File	generatenonwords.py
Dependency	This is the second python file to be executed when reading real words from the prepared text file (1 File Name: Dataset Prepare)
Python Code	*# Importing defaultdict*
	from collections import defaultdict
	import cx_Oracle
	import logging
	from os import walk
	import pathlib
	import random
	en=['a','b','c','d','e','f','g','h','i','h','k','l','m','n','o','p','q','r','s','t','u','v','w','x','y','z']
	orthographic_dict = { "ortho1" : ['ح', 'ج', 'خ'],
	"ortho2" : ['ب', 'ت', 'ث'],
	"ortho3" : ['س', 'ص', 'ش'],
	"ortho4" : ['ذ', 'د'],
	"ortho5" : ['ض','ذ','ظ'],
	"ortho6" : ['ق','ك','ف'],
	"ortho7" : ['ع','غ']
	}
	phonological_dict = { "phono1" : ['ء','ه','خ','ح','ع','غ'],
	"phono2" : ['ت','ث','ج','د','ذ','ر','ز','س','ش','ص','ض','ط','ظ','ق','ك','ل','ث','ي'],

	"phono3" : ['ب','ف','م','و'],
	"phono4" : ['أ','و','ي','ا'],
	}
	def main():
	#getNonWords();
	getNonWords_db();
	def getNonWords_db():
	ipaddress = 'localhost'
	username = 'LRT'
	password = 'Dec$$2020'
	port = '1521'
	tnsname = 'ORCLM'
	try:
	conn = cx_Oracle.connect(username + '/' + password + '@' + ipaddress + ':' + port + '/' + tnsname,
	encoding='UTF-8', nencoding='UTF-8')
	cur = conn.cursor()
	mylist=[]
	for i in range(10):
	records=cur.execute('SELECT id, WORD FROM lrt_word_frequency ')
	reconn = 0
	for result in records:

	reconn +=1
	id=result[0]
	word=result[R1]
	if len(word) < 3:
	continue
	print("result",result[0])
	print(id)
	print(word)
	print("result", result[R1])
	print("id=",id, "word=",word)
	word=str(word).replace(")","").replace("(","").re-place("'","").replace(",","")
	print(str(result).replace(")","").replace("(","").replace("'","").re-place(",",""))
	print("before calling", word)
	#generate_nonwords_ortho(id, word, conn)
	generate_nonwords_phono(id, word, conn)
	except Exception:
	logging.error("Database Connection Error")
	raise
	def getNonWords():
	ProcessedDSPath = "C:\\Users\\mohammad.nassar\\PycharmPro-jects\\storeData\\ProcessedDS"
	print(ProcessedDSPath)
	for p in pathlib.Path(ProcessedDSPath).iterdir():

```
        if p.is_file():
            if p.name.strip().endswith(".txt") and p.name.strip() == "Tokenized_All_v5.txt":
                with open(p, encoding='utf-8') as f1:
                    data = f1.readlines()
                    for line in data:
                        words = line.split()
                        if words[0].isalpha():
                            for i in words[0]:
                                if i in en:
                                    break
                            print(words[0])
                            #generate_nonwords_ortho(words[0])
                            #break
                            #generate_nonwords_phono(words[0])
                f1.close()
    f1.close()
def getOrthoList(orthochar):
    for ortho in orthographic_dict.values():
        if orthochar in ortho:
                return ortho;
def getPhonoList(phonochar):
    for phono in phonological_dict.values():
        if phonochar in phono:
```

	return phono;
	def generate_nonwords_ortho(id, ortho_word, db_conn):
	print("****************** Orthoo ****************")
	print("word", ortho_word)
	word_id=id
	print("length",ortho_word, len(ortho_word)-1)
	conn=db_conn
	print(conn)
	replace_index = random.randint(2, len(ortho_word)-1)
	print("replace_index",replace_index)
	word_list=[]
	word_list=ortho_word
	prefix = word_list[0:replace_index]
	postfix= word_list[replace_index+1 ::]
	print("prefix", prefix)
	print("postfix", postfix)
	replace_char= word_list[replace_index]
	print("replace_char=" + replace_char)
	ortho_list=[]
	ortho_list = getOrthoList(replace_char)
	print("ortho_list",ortho_list)
	print(type(ortho_list))
	if not ortho_list:

	print("list is empty")
	else:
	for ortho_char in ortho_list:
	if ortho_char != replace_char:
	new_word_list=[]
	new_word= prefix + ortho_char + postfix
	new_word_list.append(new_word)
	print("new_word_list", new_word_list)
	print("new_word", ortho_word)
	query = f"INSERT INTO lrt_nonwords (word_id, non_word, type, replace_char_index, replaced_char, prefix,ORIGINAL_CHAR, postfix) VALUES('{word_id}','{new_word}', 1,{replace_index},'{ortho_char}','{prefix}','{replace_char}','{postfix}')"
	conn.cursor().execute(query)
	conn.commit()
	print(query)
	def generate_nonwords_phono(id, phono_word, db_conn):
	print("******************** Phonooooo *****************")
	print("word", phono_word)
	word_id = id
	print("length", phono_word, len(phono_word) - 1)
	conn = db_conn
	replace_index = random.randint(2, len(phono_word) - 1)
	word_list = []
	word_list = phono_word

	prefix = word_list[0:replace_index]
	postfix = word_list[replace_index + 1::]
	replace_char = word_list[replace_index]
	phono_list = []
	phono_list = getPhonoList(replace_char)
	phono_list = getPhonoList(replace_char)
	if not phono_list:
	print("list is empty")
	else:
	for phone_char in phono_list:
	if phone_char != replace_char:
	new_word_list = []
	new_word = prefix + phone_char + postfix
	query = f"INSERT INTO lrt_nonwords (word_id, non_word, type, replace_char_index, replaced_char, prefix,ORIGINAL_CHAR, postfix) VALUES('{word_id}','{new_word}', 2,{replace_index},'{phone_char}','{prefix}','{replace_char}','{postfix}')"
	conn.cursor().execute(query)
	conn.commit()
	print(query)
	if __name__ == "__main__":
	main()
	The End

Script Name: Storing Preprocessed Dataset in the Database Schema (storeData.py)	
Process Name	storeData.py
Description	Read cleaned data produced by (1 File Name: Dataset Prepare), line by line, each line contains single word, then insert them in the database schema (structured table)
Code File	File Name: storeData.py
Dependency	Execution sequence: after readiness of prepared data set (1 File Name: Dataset Prepare)
Python Code	
	import cx_Oracle
	import logging
	class DB_CONTROL:
	def __init__(self):
	print("created")
	def INSERT_WORDS(this):
	ipaddress = 'localhost'
	username = 'LRT'
	password = 'Dec$$2020'
	port = '1521'
	tnsname = 'ORCLM'
	try:
	conn = cx_Oracle.connect(username + '/' + password + '@' + ipaddress + ':' + port + '/' + tnsname,

	encoding='UTF-8', nencoding='UTF-8')
	except Exception:
	logging.error("Database Connection Error")
	raise
	cur = conn.cursor()
	ProcessedDSPath = "C:\\\\Users\\\\mohammad.nassar\\\\PycharmProjects\\\\storeData\\\\ProcessedDS\\\\"
	with open(ProcessedDSPath + "Tokenized_All_v5.txt", encoding='utf-8') as Tokenized_File:
	data_lines = Tokenized_File.readlines()
	for line in data_lines:
	words = line.split()
	for word in words:
	try:
	cur.execute(u"INSERT INTO lrt_corpus(word) VALUES('" + word + "')")
	conn.commit()
	except Exception as e:
	content = 'not connected'
	print(e)
	Tokenized_File.close()
	cur.close()
	conn.close()
	ins = DB_CONTROL()
	ins.INSERT_WORDS()

```
    except Exception:
        logging.error("Database Connection Error")
        raise
def getNonWords():
    ProcessedDSPath = "C:\\Users\\mohammad.nassar\\PycharmProjects\\storeData\\ProcessedDS"
    print(ProcessedDSPath)
    for p in pathlib.Path(ProcessedDSPath).iterdir():
        if p.is_file():
            if p.name.strip().endswith(".txt") and p.name.strip() == "Tokenized_All_v5.txt":
                with open(p, encoding='utf-8') as f1:
                    data = f1.readlines()
                    for line in data:
                        words = line.split()
                        if words[0].isalpha():
                            for i in words[0]:
                                if i in en:
                                    break
                            print(words[0])
                            #generate_nonwords_ortho(words[0])
                            #break
                            #generate_nonwords_phono(words[0])
                f1.close()
```

	f1.close()
	def getOrthoList(orthochar):
	for ortho in orthographic_dict.values():
	if orthochar in ortho:
	return ortho;
	def getPhonoList(phonochar):
	for phono in phonological_dict.values():
	if phonochar in phono:
	return phono;
	def generate_nonwords_ortho(id, ortho_word, db_conn):
	print("****************** Orthoo ****************")
	print("word", ortho_word)
	word_id=id
	print("length",ortho_word, len(ortho_word)-1)
	conn=db_conn
	print(conn)
	replace_index = random.randint(2, len(ortho_word)-1)
	print("replace_index",replace_index)
	word_list=[]
	word_list=ortho_word
	prefix = word_list[0:replace_index]
	postfix= word_list[replace_index+1 ::]
	replace_char= word_list[replace_index]

```
    print("replace_char=" + replace_char)
    ortho_list=[]
    ortho_list  = getOrthoList(replace_char)
    print("ortho_list",ortho_list)
    print(type(ortho_list))
    if not ortho_list:
        print("list is empty")
    else:
        for ortho_char in ortho_list:
            if ortho_char != replace_char:
                new_word_list=[]
                new_word= prefix + ortho_char + postfix
                new_word_list.append(new_word)
                print("new_word_list", new_word_list)
                print("new_word", ortho_word)
                query = f"INSERT INTO lrt_nonwords (word_id, non_word,
type, replace_char_index, replaced_char,
prefix,ORIGINAL_CHAR, postfix)
VALUES('{word_id}','{new_word}', 1,{replace_index},'{or-
tho_char}','{prefix}','{replace_char}','{postfix}')"
                conn.cursor().execute(query)
                conn.commit()
                print(query)
def generate_nonwords_phono( id, phono_word, db_conn):
    print("******************    Phonooooo    ****************")
    print("word", phono_word)
    word_id = id
    print("length", phono_word, len(phono_word) - 1)
    conn = db_conn
    replace_index = random.randint(2, len(phono_word) - 1)
    word_list = []
    word_list = phono_word
    prefix = word_list[0:replace_index]
    postfix = word_list[replace_index + 1::]
    replace_char = word_list[replace_index]
    phono_list = []
    phono_list = getPhonoList(replace_char)
    phono_list = getPhonoList(replace_char)
    if not phono_list:
        print("list is empty")
    else:
        for phone_char in phono_list:
            if phone_char != replace_char:
                new_word_list = []
                new_word = prefix + phone_char + postfix
                query = f"INSERT INTO lrt_nonwords (word_id, non_word, type, replace_char_index, replaced_char, prefix, ORIGINAL_CHAR, postfix) VALUES('{word_id}','{new_word}', 2,{replace_index},'{phone_char}','{prefix}','{replace_char}','{postfix}')"
```

	conn.cursor().execute(query)
	conn.commit()
	print(query)
	if __name__ == "__main__":
	main()

Script Name: Generating N-Gram (ngram-char.py)	
Process Name	ngram-char.py
Descrip-tion	ngram-char.py
Code File	File Name: ngram-char.py
Depend-ency	Execution sequence: after readiness of prepared data set (1 File Name: Dataset Prepare)
	Word character n-gram generation is runnning here.
	all character n-grams of each single word is stored in a list in a file named "ngram1.txt"
	import pathlib
	def ngram_token(word="فلسطين"):
	grams_list = []
	gram_rank = 2
	while gram_rank < len(word):
	for i in range(len(word) - 1):
	if len(word[i:i + gram_rank]) == gram_rank:

	grams_list.append(word[i:i + gram_rank])
	gram_rank = gram_rank + 1
	grams_list.append(word)
	FileAppender1 = open("C:\\Users\\mohammad.nassar\\PycharmPro-jects\\storeData\\ProcessedDS\\" + "ngram_file.txt",
	"a", encoding='utf-8')
	FileAppender1.write(str(grams_list))
	FileAppender1.write("\n")
	FileAppender1.close()
	def readFile():
	ProcessedDSPath = "C:\\Users\\mohammad.nassar\\PycharmPro-jects\\storeData\\ProcessedDS\\"
	print(ProcessedDSPath)
	for p in pathlib.Path(ProcessedDSPath).iterdir():
	if p.is_file():
	if p.name.strip().endswith(".txt") and p.name.strip() == "To-kenized_All_v5.txt":
	with open(p, encoding='utf-8') as f1:
	data = f1.readlines()
	for line in data:
	words = line.split()
	for word in words:
	ngram_token(word)
	f1.close()
	if __name__ == "__main__":

	readFile()

Appendix 2. Project Database Source Code

Table Name: Filtered Dataset (lrt_corpus_filtered)	
Process Name	Holding clean corpus dataset
Description	Result of original dataset cleansing
	-- Generated 7/14/2020 9:19:16 AM from LRT@ORCLM
Code	CREATE TABLE lrt_corpus_filtered
	(word_seq NUMBER NOT NULL,
	word VARCHAR2(80 BYTE),
	is_diacritized CHAR(1 BYTE) NOT NULL,
	is_valid CHAR(1 BYTE) NOT NULL,
	itime DATE NOT NULL,
	attribute_1 CHAR(1 BYTE),
	attribute_2 CHAR(1 BYTE),
	attribute_3 CHAR(1 BYTE),
	attribute_4 CHAR(1 BYTE),
	attribute_5 CHAR(1 BYTE),
	attribute_6 CHAR(1 BYTE),
	attribute_7 CHAR(1 BYTE),
	attribute_8 CHAR(1 BYTE),
	attribute_9 CHAR(1 BYTE),
	attribute_10 CHAR(1 BYTE))
	SEGMENT CREATION IMMEDIATE

	PCTFREE 10
	INITRANS 1
	MAXTRANS 255
	TABLESPACE users
	STORAGE (
	INITIAL 65536
	NEXT 1048576
	MINEXTENTS 1
	MAXEXTENTS 2147483645
	)
	NOCACHE
	MONITORING
	NOPARALLEL
	LOGGING
	/

Table Name: Word Frequency (lrt_word_frequency)	
Process Name	Store calculated word frequency
Description	Considering frequency for selecting a word to generate nonwords form it.
Code	CREATE TABLE lrt_word_frequency
	(id NUMBER(9,0) NOT NULL,
	word VARCHAR2(100 BYTE) NOT NULL,

	frequency NUMBER(9,0) NOT NULL)
	SEGMENT CREATION IMMEDIATE
	PCTFREE 10
	INITRANS 1
	MAXTRANS 255
	TABLESPACE users
	STORAGE (
	INITIAL 65536
	NEXT 1048576
	MINEXTENTS 1
	MAXEXTENTS 2147483645
	)
	NOCACHE
	MONITORING
	NOPARALLEL
	LOGGING
	/
	CREATE OR REPLACE TRIGGER lrt_word_frequency_seq
	BEFORE
	INSERT
	ON lrt_word_frequency
	REFERENCING NEW AS NEW OLD AS OLD
	FOR EACH ROW

	BEGIN
	SELECT lrt_word_frequency_seq.nextval
	INTO :new.ID
	FROM dual;
	END;
	/
Table Name: Generated Nonwords Dataset (lrt_nonwords)	
Process Name	Store generated nonwords
Description	Nonwords with different dimensions as length, original word and replacement letter.
	-- Generated 7/14/2020 9:18:32 AM from LRT@ORCLM
Code	CREATE TABLE lrt_nonwords
	(id NUMBER(9,0),
	word_id NUMBER(9,0),
	non_word VARCHAR2(100 BYTE),
	type NUMBER(*,0),
	replace_char_index NUMBER(*,0),
	replaced_char VARCHAR2(10 BYTE),
	prefix VARCHAR2(80 BYTE),
	original_char VARCHAR2(10 BYTE),
	postfix VARCHAR2(80 BYTE),
	itime DATE DEFAULT sysdate,
	is_nonword VARCHAR2(5 BYTE))

	SEGMENT CREATION IMMEDIATE
	PCTFREE 10
	INITRANS 1
	MAXTRANS 255
	TABLESPACE users
	STORAGE (
	INITIAL 65536
	NEXT 1048576
	MINEXTENTS 1
	MAXEXTENTS 2147483645
	)
	NOCACHE
	MONITORING
	NOPARALLEL
	LOGGING
	/
	CREATE OR REPLACE TRIGGER lrt_nonwords_trg
	BEFORE
	INSERT
	ON lrt_nonwords
	REFERENCING NEW AS NEW OLD AS OLD
	FOR EACH ROW
	BEGIN

	SELECT lrt_nonwords_seq.nextval
	INTO :new.ID
	FROM dual;
	END;
	/
Table Name: Nonwords Generation Types (lrt_generation_type)	
Process Name	Holding nonwords generation type [phonological, orthographic]
Description	Lookup table of generation type
	-- Generated 7/14/2020 9:18:41 AM from LRT@ORCLM
Code	CREATE TABLE lrt_generation_type
	(id NUMBER(1,0),
	type VARCHAR2(15 BYTE))
	SEGMENT CREATION IMMEDIATE
	PCTFREE 10
	INITRANS 1
	MAXTRANS 255
	TABLESPACE users
	STORAGE (
	INITIAL 65536
	NEXT 1048576
	MINEXTENTS 1
	MAXEXTENTS 2147483645

	)
	NOCACHE
	MONITORING
	NOPARALLEL
	LOGGING
	/

Table Name: LRT Master Data (lrt_exam)	
Process Name	Receiving learner details
Description	Learner details that used in results analysis
	-- Generated 7/14/2020 9:18:58 AM from LRT@ORCLM
Code	CREATE TABLE lrt_exam
	(exam_id NUMBER DEFAULT 1 ,
	age NUMBER(2,0),
	gender CHAR(1 BYTE),
	score FLOAT(126),
	native CHAR(1 BYTE),
	attribute_1 VARCHAR2(20 BYTE),
	attribute_2 VARCHAR2(20 BYTE),
	attribute_3 VARCHAR2(20 BYTE),
	attribute_4 VARCHAR2(20 BYTE),
	attribute_5 VARCHAR2(20 BYTE))

	SEGMENT CREATION IMMEDIATE
	PCTFREE 10
	INITRANS 1
	MAXTRANS 255
	TABLESPACE users
	STORAGE (
	INITIAL 65536
	NEXT 1048576
	MINEXTENTS 1
	MAXEXTENTS 2147483645
	)
	NOCACHE
	MONITORING
	NOPARALLEL
	LOGGING
	/
	ALTER TABLE lrt_exam
	ADD CONSTRAINT lrt_exam_pk PRIMARY KEY (exam_id)
	USING INDEX
	PCTFREE 10
	INITRANS 2
	MAXTRANS 255
	TABLESPACE users

	STORAGE (
	INITIAL 65536
	NEXT 1048576
	MINEXTENTS 1
	MAXEXTENTS 2147483645
	)
	/
	CREATE OR REPLACE TRIGGER lrt_exam_trg
	BEFORE
	INSERT
	ON lrt_exam
	REFERENCING NEW AS NEW OLD AS OLD
	FOR EACH ROW
	BEGIN
	SELECT lrt_exam_seq.nextval
	INTO :new.student_id
	FROM dual;
	END;
	/

Table Name: LRT Test Details (lrt_exam_details)	
Process Name	Holding exam items

Description	Exam nonwords and their specifications
	-- Generated 7/14/2020 9:19:09 AM from LRT@ORCLM
Code	CREATE TABLE lrt_exam_details
	(detail_id NUMBER(4,0) ,
	exam_id NUMBER(3,0),
	template_id NUMBER(2,0),
	id NUMBER(10,0),
	word_id NUMBER(10,0),
	item VARCHAR2(20 BYTE),
	real_answer CHAR(1 BYTE),
	student_result CHAR(1 BYTE),
	type CHAR(20 BYTE),
	attribute_4 VARCHAR2(20 BYTE),
	attribute_5 VARCHAR2(20 BYTE))
	SEGMENT CREATION IMMEDIATE
	PCTFREE 10
	INITRANS 1
	MAXTRANS 255
	TABLESPACE users
	STORAGE (
	INITIAL 65536
	NEXT 1048576

	MINEXTENTS 1
	MAXEXTENTS 2147483645
	)
	NOCACHE
	MONITORING
	NOPARALLEL
	LOGGING
	/
	ALTER TABLE lrt_exam_details
	ADD CONSTRAINT lrt_exam_details_con PRIMARY KEY (detail_id)
	USING INDEX
	PCTFREE 10
	INITRANS 2
	MAXTRANS 255
	TABLESPACE users
	STORAGE (
	INITIAL 65536
	NEXT 1048576
	MINEXTENTS 1
	MAXEXTENTS 2147483645
	)
	/
	CREATE OR REPLACE TRIGGER lrt_exam_dt_trg

	BEFORE
	INSERT
	ON lrt_exam_details
	REFERENCING NEW AS NEW OLD AS OLD
	FOR EACH ROW
	BEGIN
	SELECT lrt_exam_dtl_seq.nextval
	INTO :new.detail_id
	FROM dual;
	END;
Table Name: LRT Tempates (exam_template_items)	
Process Name	Creating exam templates' items
Description	Retrieve data from nonwords and lrt_corpus_filtered to formualte exam te-maples,
Code	-- Generated 7/14/2020 9:20:13 AM from LRT@ORCLM
	CREATE TABLE exam_template_items
	(temp_id NUMBER,
	id NUMBER(9,0),
	word_id NUMBER(9,0),
	non_word VARCHAR2(100 BYTE),
	type NUMBER)
	SEGMENT CREATION IMMEDIATE
	PCTFREE 10

	INITRANS 1
	MAXTRANS 255
	TABLESPACE users
	STORAGE (
	INITIAL 65536
	NEXT 1048576
	MINEXTENTS 1
	MAXEXTENTS 2147483645)
	NOCACHE
	MONITORING
	NOPARALLEL
	LOGGING

Table Name: N-Grams Dataset (lrt_5_gram_summary)	
Process Name	Holding N-Gram dataset
Description	Result of original dataset cleansing
	CREATE TABLE lrt_5_gram_summary
Code	(frequency NUMBER,
	gram VARCHAR2(30 BYTE))
	SEGMENT CREATION IMMEDIATE
	PCTFREE 10
	INITRANS 1
	MAXTRANS 255

	TABLESPACE users
	STORAGE (
	INITIAL 65536
	NEXT 1048576
	MINEXTENTS 1
	MAXEXTENTS 2147483645) NOCACHE