# Arabic Text Correction Using Dynamic Categorized Dictionaries
## A Statistical Approach

Adnan H. Yahya and Ali Y. Salhi

Department of Computer Systems Engineering, Birzeit University, Birzeit, Palestine.
yahya@birzeit.edu, asalhi@birzeit.edu

***Abstract*— This paper describes a technique for spelling and correcting Arabic text that provides different variables that can be controlled to give customized results based on the properties of the processed text. The proposed technique depends on dynamic dictionaries controlled and customized based on the input text categorization. In the research reported here we employ a statistical/corpus-based approach with data obtained from the Arabic Wikipedia and local Palestinian newspapers. Based on corpus statistics we constructed databases of words and their frequencies as single, double and triple expressions and used that as the infrastructure for our spelling and text correction technique. Our spelling technique builds on earlier work[7], but using new spelling variables and dynamic dictionaries based on categorized texts. We briefly report on the results of preliminary testing and analysis. While the results reported here are promising, they must be viewed as work in progress, still in need of more testing, refining, integration and deployment in real life settings.**

***Keywords*** **— Natural Language Processing; Arabic Wikipedia; Arabic Text Correction; Categorized Corpus; Text Categorization**

## I. INTRODUCTION

The Arabic online content has increased from 0.1% of the worldwide content (40 million of 40 billion online pages) in 2007 to 0.3% (156 million of 47 billion pages) in 2010, an increase of 300% in three years and was expected to reach 330 million pages by the start of 2012[1]. This means that there is a global awareness among Arab writers and users about the importance of creating and publishing Arabic web content. Also this can be noticed through many online initiatives such as (on October 23, 2010) the "Digitally Open: Innovation and Open Access Forum" hosted by , ictQATAR and Creative Commons[2]. The forum addresses how innovation can push and increase sharing and openness of Arabic content. Earlier, Mohammed bin Rashid Al Maktoum Foundation in Dubai started the Sawaed program focusing on developing the capabilities of talented Arab entrepreneurs by providing non-refundable grants with a major recent focus on the development of online Arabic content[3]. Also, in 2007 the Information and Communication Technology Division (ICTD) at UN-ESCWA launched a project on "Promotion of the Digital Arabic Content Industry through Incubation"[4]. With this increase in Arabic content awareness and creation, there is a great need for tools to overcome the many challenges in processing and retrieving Arabic web content. In this paper we describe a technique for spelling and correcting Arabic text. The proposed technique provides different variables that can be controlled to give customized results based on the category of the text being processed. The technique depends on dynamic dictionaries that can be controlled and customized based on the input text categorization. We employ a statistical/Corpus-based approach built on data obtained from the Arabic Wikipedia and local Palestinian newspaper. Based on corpus statistics we constructed databases of words and their frequencies as single, double and triple expressions and used that as the infrastructure for our spelling and text correction technique.

The idea of using personalized dictionaries for spell checking to account for the peculiarities of individual writing styles and word usage is not new. However, usually the personalized dictionary adds user's words to the base dictionary to avoid tagging correct user words as errors. Our dictionary here is nontraditional in the sense that it has double and triple expressions as well as single words, plus we associate frequencies with each expression to reflect its usage and utilize this frequency to rank the candidate corrections. We further extend the idea here to allow the user to work with different dictionaries based on the topic he/she is writing on: that is, the used dictionary need not always include the base dictionary, but the latter is replaced by a specialized one that is topic-specific. Since our spellchecking offers correction suggestions as well, we also force the suggestion order to take into account the word usage in that category. The approach will use a set of weights to account for the various sources of spelling errors in Arabic to rank the candidate corrections. So, it theoretically can adapt to user behavior in terms of the dominant source of errors in his/her writing which can be used to determine the choice of the weighing parameters in the ranking equation.

Taking a look at earlier and related work, one can see that there are few available Arabic spell checkers that can be split into *commercial* such as Microsoft spellchecker used in Microsoft products and *free* such as Ayaspell which is used in OpenOffice.org (an open source multilingual office suite)[5]. We did a quick comparison between Ayaspell, Microsoft spell checkers and our approach, which is discussed in section V.

It might be that the simplest technique to create a spellchecker is by ranking a set of possible results based on the distance between them and the current investigated word, using a technique such as Levenshtein distance which works as a metric that gives the minimum number of steps needed to convert one string to another [6,7,8]. Another technique that can be used is Agrep which is based on the minimum *editing distance* between words by insertions, deletions and substitutions[6,9]. Yet another

technique used is based on Hamming distance to overcome substitution and transposition spelling errors[9]. Hamming distance approach calculates the number of positions at which the corresponding symbols are different[10]. We adopt Levenshtein distance in our spelling algorithm, but with another method that calculates the similarity between two strings based on the number of adjacent character pairs which are contained in the two strings. (Discussed in Section III).

One can talk about two types of error processing:

- Spelling errors detection: It detects which words are possibly wrong in a text. One can simply compare the text under processing with a dictionary of Arabic words, and extract all words with no possible match in the dictionary.
- Spelling errors correction: This can be classified into context dependent or context independent. If the spelling is based on information available from the processed text then it's a context dependent, else, if the method can correct individual words based on dictionaries without reference to other text parts, then it is context independent. There are many techniques used in correction based on editing distance and others based on ranking techniques such as those introduced in[7,9] where parameters such as soundex, shape similarity, keyboard keys locations and statistical frequency analysis are used to rank alternative spellings.

Our suggested approach is a mix between context dependent (using categorization) and independent (using customized dictionaries) and has three components:

- Misspelled words detection techniques: this depends on an Arabic corpus (dictionary). If a word in an Arabic input text is not found in the dictionary, then it is considered a misspelled word. It may also be considered misspelled if it is unlikely to occur in the given context (say in a given sequence of words).
- Dynamic dictionaries that can be controlled and customized based on the input text categorization; we built resources for different categories, something we will discuss in section II. The categorization can be controlled either manually or automatically through a categorization algorithm.
- Ranking metric based on a modified version of Levenshtein distance with a text similarity algorithm[11] which we will discuss in section III. There, we employ ranking parameters such as soundex, shape similarity, keyboard keys location and statistical frequency and string positioning.

The rest of the paper is organized as follows: We address building dynamic categorized dictionaries based on Arabic Wikipedia in section II. The dictionaries that will be adopted in our spelling algorithm and the different ranking and spelling variables used are discussed in section III. The algorithm also depends on a categorization technique that gives promising initial results and is highlighted in section IV. In section V the spelling algorithm is detailed with some initial testing results and complexity analysis. Then we give our conclusions and point to some future work. While the results reported here are promising, they must be viewed as work in progress, still in need of further testing, refining, integration and deployment in real life settings.

## II. SPELLING DICTIONARIES AND DATA COLLECTION

For the spelling algorithm we need to build dictionaries to be used as the base for the checking and analyzing process. The algorithm will depend on several sets of dictionaries.

### *A. General Corpus*

The first set depends on a statistical/Corpus approach based on contemporary data we obtained from various sources. The corpus has around 75 million words of written Arabic covering different topics (for more about this corpus please check[7]). Data statistics on our corpus are shown in Table I, the corpus was processed to output single, double and triple expressions. So we will have three different general dictionaries each containing words/expressions sets with frequency of appearance.

### *B. Wikipedia Categorization*

The second is a categorized set of data that depends mainly on Wikipedia. The data was collected using an automated process of connecting related articles together based on a manual categorization done by the Wikipedia editors (each article in the Wikipedia is tagged by different keywords/categories) and a list of collected Arabic Wikipedia topics obtained from the outputs of the Arabic online content indications project from the computer research institute in King Abdul-Aziz City for Science and Technology [12]. The list provides data rows of the form <title, content>. The number of documents/articles in this list is 96,128, which means the number of titles will be the same. Using these titles we can create sets of related articles and to do so we need articles to be tagged by keywords or categories and that is already done by Wikipedia editors.

Wikipedia editors use manual tagging/categorization. That is, each editor can suggest categories or keywords to tag articles, which means each article can be tagged under different categories. The overlapping categories are not fixed and one can find categories such as , رؤساء فلسطين رؤساء وزراء فلسطين , رؤساء السلطة الوطنية الفلسطينية those categories can be merged in one category such as قادة فلسطينيين or in a more general category which is أخبار فلسطين or a still the more general category سياسة. The categories that can be found in Wikipedia are too specific on one hand and on another can be repeated using different terms as seen in the above example. Figure 1 shows an article talking about القدس that has 11 categories and some of the categories are really important such as , مدن الضفة الغربية مدن ذات أهمية ,أماكن إسلامية مقدسة ,مدن كنعانية ,عواصم ثقافة عربية دينية مسيحية and so on.

TABLE I.
DATA STATISTICS

| Description | Statistics |
|---|---|
| Processed Words | 75,132,120 |
| Arabic Words (no repeat) | 962,879 |
| Arabic Words ( F > 1) [a] | 519,827 |
| Multi words expression (no repeat) | 1,843,274 |
| Triple words expression (no repeat) | 1,414,010 |
| Number of documents (PDFs , HTML) | Around 80,000 |
| Average letters per word | 5.4 letter |
| The most frequent word | 1,203,663 (في) |
| Number of letters for the longest word | 15 (الكهرومغناطيسية) |


Figure 1: Categories for the title/article "القدس"

Based on Wikipedia categories we can link articles with other articles based on the shared categories between them, the more shared categories the more the articles are connected and thus related. Also this can mean that there is a possible relation between categories in different articles if the categories appear jointly in different articles. For example if text A is categorized under, say, ميكانيكا and text B is categorized under , قوانين نيوتن , ميكانيكا طاقة حركية then we can conclude that those three categories are related. However, if we go deep in this relation analysis we may end up connecting all categories in the Wikipedia (which is not good) so one should be wise in selecting limitations in relation depth and maybe interfere manually to have control over how deep the categories/articles relation goes. We developed an approach (we named *related categories approach*) that starts with a predefined category (starting point), say فيزياء , it parses the articles to filter the ones with فيزياء as one of their categories, then for each found article, the categories found in that article are added to a queue, thus if an article has فيزياء, ميكانيكا and طاقة as categories then both ميكانيكا and طاقة are added to the queue to be parsed in the same way as فيزياء . When parsing all the articles that have فيزياء as one of its categories, we move to the next category in the queue which is for example ميكانيكا , same is done here as in فيزياء, each category may bring new categories to the queue. Each seen category will hold a variable that indicates the number of articles in that category. This variable reflects the importance of a category so if ميكانيكا was found in a large number of articles then it will be reflected in its importance (the articles may have فيزياء as one of their categories and may not, even though the starting point is فيزياء) and so on for all other categories. The question that arises here is when to stop? A variable N is introduced that control how many articles to process, for example we set N to 50, which means when the number of processed articles reaches 50 the operation stops (starting from articles that hold فيزياء as a category), and the categories stored in the queue (ranked based on their importance, which is the number of articles processed that include a certain category) are considered for manual check. A manual phase is adopted here to make sure that the categories in the queue (which is added due the parsing of the 50 articles) are truly related and do not cause major problems in categorization. Table II shows some categories with their top 10 related categories; let's take the category فيزياء for example. We parse all the articles in all the related categories for فيزياء (extracting their Arabic content), that is to parse the articles (from the 96,128 list of articles) that includes at least one of the categories related to فيزياء, in the end we will have categorized corpus for فيزياء .

Increasing N will increase the categories and thus increase the number of articles processed, but will increase the need for manual checking of the added categories in order to maintain control over the quality and value of the corpus.

TABLE II.
SOME CATEGORIES WITH THEIR TOP 10 RELATED CATEGORIES, USING THE RELATED CATEGORIES APPROACH

| **Selected word to find related categories to: فيزياء** | |
|---|---|
| فيزياء | علم الكون |
| نسبية | أعداد الكم |
| صواريخ | ضوء |
| نظريات فيزيائية | أمثلة كونية |
| إلكترونيات | اتصالات |
| **Selected word to find related categories to: طب** | |
| طب | صحة |
| علم الأدوية | منظمة الصحة العالمية |
| أمراض | أمراض وراثية |
| مصطلحات طبية | مضادات حيوية |
| صيدلة | علم الوراثة |
| **Selected word to find related categories to: علم حاسوب** | |
| علم الحاسوب | حوسبة |
| شبكات عصبونية | برمجة |
| تنقيب البيانات | إنترنت |
| تحليل رياضي | معلوماتية عصبونية |
| أمن المعلومات | أمن شبكة الحاسوب |
| **Selected word to find related categories to: دين** | |
| دين | فقه إسلامي |
| فقه عبادات | ديانات آسيا |
| أركان الإسلام | جهاد |
| نصوص دينية | كونفوشيوسية |
| شريعة إسلامية | طوائف إسلامية |

Using this technique we built a Wikipedia based categorized corpus. Table III gives statistics about the predefined categories that we adopted; of course the data in this corpus is subject to change due continuing data processing. Those categories will be used in experiments related to spelling by first categorizing the text being spellchecked in order to give a higher rank to possible results within the text category. For example, the following sentence هذا ريال مجديد if it comes in a sport text then مجديد most likely to be مدريد but if it comes in a financial text then مجديد is most likely to be جديد.

So far, we defined 26 categories as seen in Table III. Categories: سباق سيارات, كرة قدم, كرة سلة, تنس, أولمبياد can all be joined under رياضة category. This will give more dynamic options of using a major category رياضة if the text is about رياضة or a subcategory كرة قدم for example if the text is about رياضة in general and كرة قدم in particular.

Working with a more specific category like كرة قدم will help in decreasing the dictionary size and will improve access time (less processing and less queue size) compared with more general category like رياضة.

Using the "related categories" approach we can create new categories by starting from the desired category. For example if a user decides that there is a need for the category الطيور then adopting the same steps we did for فيزياء will output a new corpus specialized in الطيور from the Wikipedia articles.

TABLE III.
WIKIPEDIA CATEGORIES CURRENT STATISTICS

| Category | # of unique words | # of all words processed | Average Frequency | Average length of words |
|---|---|---|---|---|
| كرة قدم | 11,035 | 54,674 | 9.45 | 5.94 |
| كرة سلة | 8,155 | 44,479 | 10.43 | 5.75 |
| تنس | 9,419 | 65,227 | 12.07 | 6.08 |
| سباقات سيارات | 5,837 | 24,477 | 7.76 | 5.78 |
| أولمبياد | 12,300 | 65,445 | 10.84 | 6.03 |
| إقتصاد | 14,405 | 52,276 | 7.55 | 6.15 |
| إسلامي | 29,219 | 65,210 | 8.41 | 5.80 |
| مسيحي | 15,329 | 19,630 | 6.57 | 6.03 |
| كهربائية وإلكترونية | 7,335 | 27,771 | 6.94 | 6.00 |
| ميكانيكية | 8,216 | 28,796 | 6.27 | 5.98 |
| كمبيوتر و شبكات | 13,945 | 62,020 | 7.63 | 6.06 |
| كيمياء | 11,717 | 45,025 | 7.26 | 6.12 |
| فيزياء | 5,204 | 16,217 | 5.88 | 5.85 |
| رياضيات | 6,928 | 26,746 | 6.82 | 6.03 |
| أحياء | 9,416 | 25,406 | 5.27 | 5.90 |
| طب بشري | 13,720 | 53,923 | 7.44 | 5.67 |
| صيدلة | 14,230 | 45,322 | 4.62 | 6.16 |
| تاريخ حديث | 20,217 | 68,985 | 7.95 | 5.96 |
| تاريخ قديم | 22,890 | 83,103 | 7.21 | 5.90 |
| شعر وأدب | 15,666 | 35,166 | 4.72 | 5.52 |
| موسيقى وغناء | 12,998 | 38,196 | 5.63 | 5.74 |
| سينما ومسرح | 14,252 | 41,662 | 5.41 | 5.82 |
| ديانات أخرى | 10,393 | 25,916 | 5.14 | 5.88 |
| سياسة | 11,2367 | 26,605 | 8.23 | 5.81 |
| موضى ومرأة | 7,318 | 14,923 | 4.32 | 5.66 |
| جغرافيا | 10,157 | 31,155 | 5.55 | 6.07 |

## III. SPELLING VARAIBLES

Our Spelling algorithm depends on different variables, such as Levenshtein distance and similarity calculations to decide on the best possible matches for a certain word, then other variables such as soundex, keyboard location, shape similarity and frequency are considered to rank the candidate matches in the most efficient way. The success measure is to have the user intended word the highest ranked suggestion. Of course those variables will be used to rank candidate matches whether they are single, double or triple expressions (based on single, double and triple dictionaries).

### *A. V1: Levenshtein distance (Lev)*

Levenshtein distance works as a metric for the minimum number of steps needed to convert string A to string B[8], A and B could be single, double or triple expressions. For example if we have A: الوظن and B: الوطن then Levenshtein distance between A and B will be 1, also C: الوزن has a difference of 1 from A, plus the word D: الوتر has 2 differences. Word D will have less ranking (based on Levenshtein distance alone) since it differs by two letters. From this last point the problem of swapping errors shows up in the normal Levenshtein distance algorithm, for example let's take A: الطون and B: الوطن then the words will have two units of distance not one. However if the user meant الوطن the system will end up ranking the latter in lower positions, but it will rank C: الطوق in a higher position due to one difference. So a modification by mapping the two letter swap differences to one unit of distanced is needed.

Levenshtein distance function will output the result based on Equation (1)

$$Lev(A,B)= 1 – [ \#ofDiffrenentLetteres/min(Length\ A, Length\ B)] \quad (1)$$

For Example assume A is البيت and B is البيثث , then the number of different letters are 2 (the last two letters in B) thus the equation will become 1-[2/min(5,6)] which equals to 0.6.

### *B. V2: Letter Pair Similarity (LPS)*

The letter pair similarity function finds out how many adjacent character pairs are contained in two strings A and B. It is most useful when we are spelling based on double and triple expressions, Equation (2) shows how the letter pair similarity is calculated[11].

$$LPS\ (A, B) = (2*|\ pairs\ (A) \cap pairs\ (B)|) / (|pairs\ (A)| + |pairs\ (B)|) \quad (2)$$

For example let us take the following strings, A: العربي الوطن and B: الوطن العربي , maybe the user really meant B not A. Using Levenshtein distance will give 10 differences between the two strings which will discard (very low ranking) expression B from the possible matches. However the letter pair similarity function will give a 100% match between both sentences based on the following calculations:
String A الوطن العربي will be divided into pairs of letters (after splitting the string based on space character) : P1{ العربي} and String B الوطن will be divided into P2{ ال, لع, عر , رب, بي , ال, لو, وط, طن} , according to (2) the similarity will be: 18/(9+9) = 1.00 which will rank the sentence high in the possible match list.

### *C. V3: Shape Similarity*

A function which measures the similarity in shape between two strings A and B. For example, if A is الوظن and the spelling algorithm possible matches has two possibilities B الوطن and C الوزن . Both are correct alternatives. However the shape similarity function will detect that B looks closer in shape to A than C since ط and ظ have almost the same shape, and letter ز doesn't look as close.

The algorithm depends on dividing the alphabet into sets of letters with related shapes then comparing strings A and B to calculate the number of related letters. If letter i in String A and letter i in String B are in the same shape group then they are considered related, Equation (3) shows how to calculate the shape similarity. For more details please check [7].

$$ShapeSimilarity(A, B) = \#OfRelatedLetters/\ max(length\ A, length\ B) \quad (3)$$

### *D. V4: Location Parameter*

A function that takes into consideration the locations of letters on the keyboard. It acts similar to the shape similarity function; however the related letter groups are based on adjacency on the keyboard. For example, the letter ا will have the following related group: { ت ل ف غ ع ة لآ أ لأ لإ إ ، ـ آ ى} which are the letters located around it in the keyboard (with Shift Key on and off), this also can be extended to restore Arabic text entered in Latin due to

failure to switch keyboard entry language (currently PC keyboards are adopted, though it can be extended to other layouts), for example we can extend the related letters group for ا to include {h, H}, which are the letters in Latin that shows up due to failure to switch keyboard entry language.

The function defines two variables 1) *equalLetters* if the two letters (Ai and Bi) are equal and 2) *relatedLetters* if the two letters are related on the keyboard based on related letters map. Equation (4) shows how to calculate the location similarity. For more details please check [7].

LetterLocation(A,B) = (#equalLetters + #relatedLetters]) / max(length A, length B) (4)

### *E. V5: Soundex Function*

Originally, this is a "phonetic algorithm for indexing words by sound as pronounced in English"[13]. In our case indexing strings by sound as pronounced in Arabic, it's similar to the other similarity functions with "related group" letters. For example the letter س has the following group that sounds like it {ث, ص, س } which are the letters with the same sound at least in some common pronunciations. Equation (5) shows how to calculate the shape similarity. For more details please check [7].

Soundex (A, B) = (#relatedLetters / max (length A, length B)) (5)

### *F. V6: String Ranking and Frequency (R&F)*

Another variable to consider in ranking the results of a possible match list is the candidate string rank (we say string because this is applied to single words, double and triple expressions) and frequency variables. String ranking refers to the position of a string in a dictionary according to frequency, and the frequency refers to the number of appearances of a string over the sum of all frequencies in the candidates match list. For example, in a list (dictionary) consisting of six strings and their dictionary frequencies { A:100 , B:75 , C:75 , D:30 , E:28 , F:10}, B for example has a rank 2 out of 5 not 6 because B and C have a ranking of 2 repeated, so the ranking value will be (2/5), the frequency of appearance of B is 75 and the string percentage frequency will equal 75/(100+75+75+30+28+10). Equation (6) shows how the string ranking and frequency variable is calculated.

R&F(s) = [Rank(s)/TotalRank]*[Freq(s)/TotalDicFreq] (6)

In section V we will explain how the various variables are combined to build the spelling algorithm.

## IV. CATEGORIZING METHOD

As we discussed in section II, categorized dictionaries are built to be used in our spelling technique. The categorizing step is done before the spelling to detect the category of an input text and provide the best categorized dictionary to be used by the spelling algorithm. The categorization process itself may be manual; the user may define the category he/she is using and the system will adopt this choice or it may be an automated process. Many approaches to categorizing text exist in the literature, and generally any of them can be used for the input text categorizing process. Here, we will demonstrate some of our work on building categorizing systems. Please note that the work addressed in this section is still in progress.

### *A. Percentage and Difference Categorization Algorithm*

The algorithm focuses on the relation between ratios in the input text words and the corresponding ratios in the reference texts (Wikipedia categories) to decide to which category to assign each word in the input text. This means it will calculate the percentage of each word in the input (words frequency/total words) and compare it with the word percentage in each category (if it exists), then find the difference between the two values and assigns the word to the category with smallest difference. For example if a word A has frequency 7 in the input text, and the size of the text is 300 words, then the percentage of A in the input table is 7/300 = 0.023333 then A percentage value is calculated in each categorized dictionary (if it exists), for example A has a frequency of 500 in a dictionary X that has a total frequency (of all words) summation of 10,000, then A in X has 500/10000 = 0.05 , then the relation between A in X and A in the input text will be the absolute value of (0.023333 – 0.05) which is 0.026667, this is done for all dictionaries and the (category) dictionary with minimum difference is assigned to A.

This process is done for all words, after removing stop words from input text (Stop words is a list of very common words which are filtered prior to/after processing of natural language data [14]) using stop words filtration method discussed in [7].

The input word and its best match category are stored in a table as <Word , Category>, then the algorithm will detect the most frequent category and consider it as the best match for the current input text. Of course in practice the chosen category will be shown to the user to make sure he/she does agree, the possibility to change the category is always available to the user.

### *B. Testing the Percentage and Difference Categorization Algorithm*

The testing was done on a sample of 380 files, which were distributed among different categories. Table IV shows the categories which were considered in the test, with files numbers and sources (web sites).

The Results of the test is shown in Table V, which shows a percentage of the successful hits.

Taking a look at earlier studies, we can make a quick judgment about how good our results are. Based on [15] experiments which evaluated the performance of two popular classification algorithms (SVM and C5.0) on categorizing Arabic text using seven Arabic corpora, the average results was 68.65% for SVM and 78.42% for C5.0. The study using Naïve Bayes algorithm reported 68.78% accuracy [16] and another study that uses kNN algorithm reported 96% accuracy results based on six categories which are :- سياسة, اقتصاد, رياضة, صحة-طب , صحة- نبات –زراعة ,سرطان [17].

It is difficult to compare our results with others because different data sets and number of categories are used in different algorithms and tests, however according to some earlier work our results are promising.

TABLE IV
TEST SAMPLE SOURCES

| Category | # of files | Web Sites |
|---|---|---|
| كرة قدم | 40 | http://www.mbc.net<br>http://www.yallakora.com/<br>http://www.kooora.com/ |
| كرة سلة | 40 | http://www.as7apcool.com<br>http://www.syrian-soccer.com<br>http://www.kooora.com/<br>http://www.yallakora.com/ |
| سباقات | 40 | http://www.bbc.co.uk<br>http://www.yallakora.com/ |
| هندسة كهربائية وإلكترونية | 40 | http://olom.info/<br>http://www.alhandasa.net<br>http://aafaq.4t.com/components.htm |
| فيزياء | 40 | http://www.physicsacademy.org/<br>http://hazemsakeek.com/<br>http://phys.olom.info/<br>http://www.schoolarabia.net<br>http://www.marefa.org |
| كيمياء | 40 | http://www.ksa-teachers.com/<br>http://www.schoolarabia.neta7ya2<br>http://www.bytocom.com/ |
| أحياء | 40 | http://www.sehha.com<br>http://www.asnanak.net<br>http://www.csmc.edu/ |
| كومبيوتر وشبكات | 40 | http://www.boosla.com/<br>http://www.bramjnet.com |
| اقتصاد | 40 | http://www.aljazeera.net<br>http://www.bbc.co.uk/arabic/<br>http://ara.reuters.com/news/business |
| سياسة | 20 | http://www.maannews.net |

TABLE V
TEST RESULTS

| # | Category | #Files | Percentage & Difference Based Algorithm results (number of Files) | Success Percentage |
|---|---|---|---|---|
| 1 | كرة قدم | 40 | 38 | 95% |
| 2 | كرة سلة | 40 | 38 | 95% |
| 3 | سباقات | 40 | 37 | 92.5% |
| 4 | هندسة كهربائية | 40 | 33 | 82.5% |
| 5 | فيزياء | 40 | 34 | 85% |
| 6 | كيمياء | 40 | 34 | 85% |
| 7 | احياء | 40 | 32 | 80% |
| 8 | كمبيوتر و شبكات | 40 | 31 | 77.5% |
| 9 | اقتصاد | 40 | 36 | 90% |
| 10 | سياسة | 20 | 16 | 80% |
| | Average | 86.58% | | |

## V. SPELLING ALGORITHM

The spelling algorithm is based on the spelling variables we addressed in section III and it is an extension of earlier work we did[7] with new variables and methods.

### *A. The Suggested Spelling Approach*

The algorithm starts by detecting the category of the current input text as just outlined to decide on which dictionary set to use. The dictionaries, including the general one (which can be used if the categorizing phase is skipped), hold single, double and triple expressions. Why have multiple expressions? Let's take the following example: the spelling of الوظن الغربي , the word الوظن is wrong and the word الغربي is correct , however if spelling each alone the word الوظن will be spelled either الوطن or maybe الوزن . That depends on the ranking system, but in the case of double expressions spelling, the system will look at the expression as one block and detect that the best solution is الوطن العربي neither الوطن الغربي nor الوزن الغربي. Here, the double expressions were useful; same is said about the triple expressions.

The spelling algorithm takes into considerations the three tables (single, double and triple expressions) when spellchecking a text. It follows the user input while typing, for example when the user types a word (w) in a sentence, the spelling algorithm checks the current {w || w, w -1 || , w, w-1, w -2} and then { w || w , w+1|| w, w+1, w+2} when possible, that is a window of one, two , or three words in both direction (to check if the error is better corrected using double or triple expressions first). For example, assume that the user is typing the following sentence : " وتتساقط الأمطار على سواحل الذهر الأبيض المتوسع " if the system doesn't take a window of three words (triple expression checking) then two words (double expression checking) before checking single words errors, it might correct the word الذهر to النهر and keep the word المتوسع as is, however if the user still writing and reaches الذهر , the system will check {على سواحل الذهر } if no possible match it will check { سواحل الذهر } if no possible match it then checks {الذهر} and may correct it to النهر. However when the user continues writing the word الأبيض, the system again will check { الذهر الأبيض } and assume it is corrected to البحر الأبيض , then the user continues writing المتوسع then the system will check { الذهر الأبيض المتوسع } and correct the whole sentence to البحر الأبيض المتوسط. If the system couldn't find a match with triple expressions; it goes back to the double expressions match. In all cases the system will check previous words, then next words for a certain error word (when it's possible to do so, that is after the user continues writing).

How the spelling algorithm works? After selecting the dictionary, the algorithm calculates *Levenshtein distance* (V1) and *Letter Pair Similarity* (V2) between each word/expression in the input text and in the dictionary, then each word/expression is assigned a value of W = V1*V2. After finishing this step the algorithm sorts the dictionary words/expressions based on their W value and the highest N are considered (N value depends on the user and we currently select the highest 20 different values), the words with their W values are stored as the possible match list in order to be ranked based on the rest of the spelling variables (V3 ~ V6).

For each word/expression in the possible match list the values of Shape Similarity (V3), Location Similarity (V4), Soundex Function (V5) and Strings Ranking & Frequency (V6) are calculated and a final value of all those variables is given to each word based on Equation (7).

$$V(Wi) = A*V3 + B*V4 + C*V5 + D*V6 \quad (7)$$

Where A, B, C, D are percentages with summation of 100% (weights). Consider A = 0.20 and B = 0.25 and C= 0.05 and D = 0.5 then the Equation will be:

$$V(Wi) = 0.20*V3 + 0.25*V4 + 0.05*V5 + 0.5*V6$$

The chosen values for A, B, C and D are not necessarily the best. They are based on experimentation and thus need more testing to decide the best range (or values) for them; Table VI shows some sample results.

We suggest an auto technique to keep changing the

weights based on user spelling behavior which is discussed next.

A question that may arise here: is why we are processing the words in two phases (calculating the distance using Levenshtein and Similarity, and then ranking)? Simply because it's better to rank words with minimum differences from the input word rather than ranking all the words in the dictionary, in the end we are looking for the word with minimum distance and high ranking. For example assume we have the word A: المفكروون if we considered two stages then the word المفكرون will be in the output , but if we did it in only one stage then the words المفسرون , المفترون will also show up.

We did a quick test to compare our sample results with results from Ayaspell and Microsoft spellers; Table VII shows the quick comparison.

TABLE VI
SAMPLE RESULTS

| # | Input | Output(s) |
|---|---|---|
| 1 | الحكونة | **الحكومة** , المكونة , الحنونة , |
| 2 | الخزيرة | **الجزيرة** , الخطيرة , الخميرة , |
| 3 | الفامويس | **القاموس** , الفانوس , الفاخوري , |
| 4 | يظحكون | **يضحكون** , يحكون , يشككون , |
| 5 | المديخ | **المديح** , المدين , المدير , |
| 6 | المدرسع | **المدرس**, المدرسة , |
| 7 | النخر الابيظ المتوشط | **البحر الأبيض المتوسط** |
| 8 | ويتعاولنون | **ويتعاونون** , ويتعاملون , ويتناولون , |

TABLE VII
SAMPLE RESULTS COMPARISON

| # | Input | Our Approach | Microsoft (Office 2010) | AyaSpell |
|---|---|---|---|---|
| 1 | الحكونة | **الحكومة** , المكونة , الحنونة , | **الحكومة ,** الحنونة , المكونة | **الحكومة** , المكونة , الحنونة, |
| 2 | الخزيره | **الجزيرة** , الخطيرة , الخميرة , | الخزيره, الخنزيرة | الخزيرة, الحريزة, **الجزيرة** |
| 3 | الفامويس | **القاموس** , الفانوس , الفاخوري , | القاموسي, **القاموس** , الفموي | **القاموس,** الفتوا, الفقوا |
| 4 | يظحكون | **يضحكون** , يحكون , يشككون , | **يضحكون,** يحكون, يمحون | **يضحكون,** يحكون, يمحكون |
| 5 | المديخ | المدير , **المديح** , المدين , | ----- | **المديح**, ألمديح, الخديم |
| 6 | المدرسع | , المدرسة, **المدرس** | **المدرس**, المدرع, المدرسة | المدرس, المدرع , المدرسا |
| 7 | النخر الابيظ المتوشط | **البحر الأبيض المتوسط** | ----- | ----- |
| 8 | ويتعاولنون | **ويتعاونون** , ويتعاملون , ويتناولون , | **ويتعاونون** | **ويتعاونون ,** وليتعاونون . ويتعاونوا |

Let's take the word الخزيرة in Table VII for example. In our approach the first results will be الجزيرة which is what we really meant by the error word. However Microsoft Office 2010 Speller didn't output الجزيرة only الخزيرة the same word with ة instead of ه , AyaSpell gave الجزيرة as an output however its ranked it 3 in its possible outputs, it seems that the word الخزيرة was a name of an old food according to [18]. Checking our general dictionary we found that the word do exist however with a very small frequency (F=2), which affected its ranking in our spelling algorithm.

We can notice also that Microsoft Office 2010 failed to retrieve a result for the word المديخ , also it seems that the word المِدّيخُ means العزيز العظيمُ [19]. With ّ on the د , however such word is not a frequent one.

We also can notice that both Ayaspell and Microsoft Office 2010 speller doesn't take double and triple expressions in processing, which give our spelling approach an advantage.

### *B. Adapting to user errors patterns: ( Auto Learning apprach)*

As seen in equation (7), the values of the A, B, C and D can be customized based on the use of the spelling algorithm. That is if the spelling is done over a text extracted from an OCR system then it is reasonable to give the shape similarity variable (addressed through A in equation (7) ) more weight since the errors in the output of an OCR system is most likely related to letters/symbol shapes. However, if the spelling algorithm will be used to fix speed typing errors then it is reasonable to give the location variable (addressed through B in Equation (7)) more weight. The point is that the system can be customized based on some initial information, given or calculated, on the environment of use.

In most cases the spelling system will not be tuned or customized manually, in normal use the errors may vary from keyboard errors to soundex or shape errors... etc.

We now introduce a technique for customizing the weights (A, B, C and D) in the ranking equation (Equation 7) of our spelling algorithm based on the user typing behavior. The technique is based on auto learning from users' errors. Assuming the user made error X: السبر and the system gave two possible outputs Y1: الصبر and Y2: الحبر , assuming the user selects Y1 as the correct output, the system will recheck the ranking variables with the correct output that is to recheck the value of V3 ~ V6 between X and Y1 and stores the result. This is done every time the user makes a mistake, the variable with the best results is the variable with the most effect on the ranking equation thus the system will increase its weight and lower the weights of other variables with smaller results. Table VIII explains how this is done.

TABLE VIII
WEIGHTS AUTO CHANGING

| X | Y | Shape Similarity (V3i) | Location Similarity (V4i) | Soundex Function (V5i) | Ranking & Frequency (V6i) |
|---|---|---|---|---|---|
| i=1 | | | | | |
| **السبر** | **الصبر** | 0.750 | 1 | 1 | 0.864 |
| Total Sum(S) | | S= S+(V3i+V4i+V5i+V6i) = 3.614 | | | |
| Ranking Weights = | | A= ∑(V3i)/S **0.208** | B= ∑(V4i)/S **0.277** | C= ∑(V5i)/S **0.277** | D= ∑(V6i)/S **0.239** |
| i=2 | | | | | |
| **كالب** | **قالب** | 0.688 | 0.688 | 1 | 0.051 |
| Total Sum (S) | | S = 3.614 +(V3i+V4i+V5i+V6i) = 6.040 | | | |
| Ranking Weights = | | A=**0.238** | B=**0.279** | C=**0.331** | D=**0.152** |

Table VIII shows an example of two mistakes (i=1, i=2) , the first one X = السبر and the user selects Y = الصبر from the output list as the correct word , the system

calculates the variables for Y then sum all of them and calculates new percentages by dividing each variable value on the total sum.

For i=2 , X= كالب and the user selects Y = قالب the system calculates the new variables and then add them to the values of the variables in the case of i=1, this is also done to the sum and then finds the new percentages.

As seen in Table VIII the soundex weight increased noticeably because the user choice was based on outputs pointing to errors related by sound. Also this can be used as a history table for the user selected outputs which can help if the user did the same mistake again; the system checks the user history before checking with the dictionary.

### *C. Initial Tests*

An initial test was done using 100 articles from the 380 articles used in testing the categorizing algorithm, (section IV). The articles were tested twice, the first time by adding manual errors; that is we manually introduced mistakes inside each article, the second time a random letter changing automated system was used to generate errors inside each article. Table IX shows the results, gives an indication about the early tests and the pass percentages (no auto learning technique is used in the tests). The tests are made with A=0.2, B=0.25, C=0.05, D=0.5.

From the results in Table IX we can conclude that adding categorized dictionaries will improve the results and will speed up the system (since we are using shorter lists of reference words which means a smaller dictionary).

In Table X we show some complexity and response time results for our algorithms (Percentage and Difference Categorization Algorithm and the Spelling Algorithm), which are used in the spelling mechanism based on dynamic categorized dictionaries.

It is worth mentioning that the process of selecting the spelling dictionary and the values of the weights can be highly customized by the user. Future work will focus on creating different types of tests in order to decide what are the best values to use for each parameter, when to adopt auto learning process and what are the best dictionaries to use and when, as well as the best-selling databases structures which are most likely to improve the performance of the entire algorithm.

TABLE IX
EARLY TESTS

| # | Test Type | Dictionary | Pass Percentage |
|---|---|---|---|
| 1 | Speed Writing (100 articles) | Auto Categorized | 82% |
| | | General | 77% |
| 2 | Auto generated errors (1 and 2 errors per word possible) (100 articles) | | |
| | | Auto Categorized (1 letter error) | 90% |
| | | Auto Categorized (2 letters error) | 83% |
| | | General (1 letter error) | 88% |
| | | General (2 letters error) | 80% |

TABLE X
COMPLEXITY AND RESPONSE TIME

| **Percentage and Difference Categorization Algorithm** | |
|---|---|
| Complexity of the algorithm | O(M*N). Where M is the number of words in the input text and N is the categorized dictionary size |
| Average Response[1] (Time on Average Machine[2]) | 0.102 Seconds |
| **Spelling Algorithm** | |
| Complexity of the algorithm | O(M*N) . Where M comes from the complexity calculations of the ranking variables and N is the dictionary (in use) size |
| Average Response[3] Using : (Time on Average Machine[2]) | |
| General Dictionary of 190,000 words with frequency >9 (not categorized) | 3.160 Seconds for each word |
| Categorized Dictionary of around 10,000 words | 0.132 Seconds for each word |

[1]Average time was calculated by testing the time of the 380 testing file.
[2]Average Machine: Intel(R) Core(TM) Due CPU, P8400 @ 2.26 GHZ & 2 GB of RAM, OS: Microsoft Windows XP(TM).
[3] Average time was calculated by testing 100 misspelled words.

## VI. CONCLUSION

We presented a spelling approach that is supported by dynamic categorized dictionaries, customized ranking variables, with the aid of a categorizing approach and auto learning spelling mechanism. In our research we employed a statistical/Corpus-based approach and data obtained from the Arabic Wikipedia and Palestinian newspaper. We described how the corpus was built especially the categorized Wikipedia corpus. Based on corpus statistics we constructed databases of words and their frequencies as single, double and triple word expressions and customized dictionaries based on automated Wikipedia articles filtration. Then we used those dictionaries as the infrastructure for our spelling and error correction method. Our spelling technique is based on earlier work but incorporating new spelling variables and dynamic dictionaries. We briefly reported on the results of preliminary testing, both on categorization and spelling. While the results reported here are promising, they must be viewed as work in progress, still in need of more testing, refining, integration and deployment in real life settings and evaluation by users in a real working environment. There will also be a need to work on better data structures to improve the performance of the tools to allow for more transparent integration into working systems.

ACKNOWLEDGEMENTS: They authors would like to thank Google and Birzeit University for supporting this research through grants, and the two anonymous referees for their comments that helped improve the paper.

## REFERENCES


[1] Veecos, Arabic online content: web metric study, Veecos Website .[Online].Available:http://www.veecos.net/portal/index.php?option=com_content&view=article&id=5997:2011-04-16-08-53-48&catid=42:doctorah&Itemid=180,[Jan, 2012].

[2] Supreme Council of Information and Communication Technology, Arab Digital Content, ictQATAR Website.[Online].Available: www.ictqatar.qa/output/Page2039.asp/,[Mar, 20,2011].

[3] Mohammed bin Rashid Al Maktoum Foundation, Sawaed Programme, Mohammed bin Rashid Al Maktoum Foundation Website.[Online].Available: http://www.mbrfoundation.ae/English/Entrepreneurship/Pages/Sawaed.aspx /,[Mar, 20,2011].

[4] ESCWA, Digital Arabic Content, ESCWA Website.[Online].Available: http://www.escwa.un.org/divisions/projects/dac/index.asp/,[Mar, 20,2011].

[5] Shaalan, K., Aref, R. and Fahmy, A. : "An Approach for Analyzing and Correcting Spelling Errors for Non-native Arabic learners."; *The 7th International Conference on Informatics and Systems (INFOS 2010);*Cairo, Egypt.; 28-30 March 2010. Website.[Online].Available: http://ieeexplore.ieee.org/xpl/freeabs_all.jsp?arnumber=5461784&abstractAccess=no&userType=inst

[6] Hodge, V. J. and Austin, J. : "A Comparison of Standard Spell Checking Algorithms and a Novel Binary Neural Approach."; *IEEE Transactions On Knowledge And Data Engineering*, Vol. 15, No. 5, September/October 2003

[7] Yahya, A. and Salhi , A. : " Enhancement Tools for Arabic Web Search : A Statistical Approach"; *7th International Conference on Innovations in Information Technology*; Abu Dhabi, United Arab Emirates.; 25-27 April 2011. Website.[Online].Available: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=5893871&isnumber=5893793

[8] Wikipedia, Levenshtein Distance, Wikipedia Website.[Online].Available: http://en.wikipedia.org/wiki/Levenshtein_distance/,[Feb, 2011].

[9] Hodge, V. J. and Austin, J. :"A Novel Binary Spell Checker."; *Artificial neural networks--ICANN 2001: International Conference*, Vienna, Austria, August 21-25, 2001. Website.[Online].Available:http://books.google.ps/books?id=YT4mrcAx6lEC

[10] Wikipedia, Hamming distance, Wikipedia Website.[Online].Available: http://en.wikipedia.org/wiki/Hamming_distance/,[Feb,2011].

[11] Catalysoft, How to Strike a Match , catalysoft Website.[Online]. Available: http://www.catalysoft.com/articles/StrikeAMatch.html

[12] The Arabic online content indications project , Computer Research Institute , King Abdul-Aziz City for Science and technolog. Website.[Online].Available: http://cri.kacst.edu.sa/en/cri-products/current-projects

[13] Wikipedia, Soundex, Wikipedia Website.[Online].Available: http://en.wikipedia.org/wiki/Soundex,[Feb,2011].

[14] Wikipedia, Stop Words, Wikipedia Website.[Online].Available: http://en.wikipedia.org/wiki/Stop_words,[Feb,2011].

[15] Al-Harbi, S., Almuhareb, A., Al-Thubaity, A., M., Khorsheed S. and Al-Rajeh, A. : "Automatic Arabic Text Classification". In: Proceedings of The 9th International Conference on the Statistical Analysis of Textual Data, Lyon-France. [Online]. Available: http://eprints.ecs.soton.ac.uk/22254/1/Arabic-Classification.pdf

[16] El-Kourdi, M., Bensaid, A. and Rachidi, T. .: "Automatic Arabic Document Categorization Based on the Naïve Bayes Algorithm". 20th International Conference on Computational Linguistics. August, 2004, Genev. [Online].Available: http://acl.ldc.upenn.edu/W/W04/W04-1610.pdf

[17] Al-Shalabi, R., Kanaan, G., and Manaf, G. H. : "Arabic Text Categorization Using kNN Algorithm". Proceedings of The 4th International Multi-conference on Computer Science and Information Technology, Vol. 4, Amman, Jordan, April 5-7, 2006. [Online].Available: http://www.uop.edu.jo/download/research/members/CSIT2006/vol4%20pdf/pg20.pdf

[18] Islamweb, Fath Al-Bari, Saheeh Bukhari Explanation. [Online].Available: http://www.islamweb.net/newlibrary/display_book.php?idfrom=9866&idto=9867&bk_no=52&ID=3017,[Feb,2012].

[19] Almaany, Detentions , [Online].Available: http://www.almaany.com/home.php?language=arabic&lang_name عربي =&word= المديخ &type_word=0, [Feb, 2012].