Arabic Numerals Checker: Checking Agreement between Numerals and Counted Objects in the Arabic Language

AHMAD T. AL-TAANI*
Department of Computer Science,
Faculty of IT and Computer Science,
Yarmouk University,
Irbid, Jordan
ahmadta@yu.edu.jo
http://faculty.yu.edu.jo/AltaaniA

SANA A. WEDIAN† AND OMAR M. DARWISH‡
Department of Computer Science,
Jordan University of Science and Technology,
Irbid, Jordan
†sana_wedian@hotmail.com
‡omar.darwish@gmail.com

Arabic Numerals are one of the major aspects to be considered in Arabic language processing. In this paper, we present an approach for developing a numeral checker application for the Arabic language, called “Arabic Numerals Checker”. Arabic Numerals Checker is intended to help users achieve competence in writing according to the Arabic numeral rules. It is supported by an explanation facility that provides the user with a clear explanation of the output. The use of the Arabic Numerals Checker, when integrated with other language processing applications, can enhance the quality of the text for users who read or write in the Arabic language. The Arabic Numerals Checker is implemented as a simple Finite State Transducer that maps numbers to the corresponding numerals in a way that respects the gender agreement feature. The proposed approach covers all the numbers in the range 1 to 999,999,999,999. Tests have been executed using random numbers in the range above. Experimental results have demonstrated that the proposed approach is promising in the sense that it provides efficient and error-free transformation.

Keywords: Arabic language processing; numeral rules; finite state transducers.

*Corresponding author.
1. Introduction

The Arabic language has a complex morphology and grammar. The complexity of the grammar rules make it difficult for most average speakers to understand and memorise all of them. Moreover, in the modern Arabic-speaking world, classical Arabic notation has been replaced with modern Arabic notation, with the result that many native speakers of Arabic are not familiar with the grammar of their own language [1].

The Arabic language has some unique features that distinguish it from other languages and make it very difficult to handle on a computer: (a) The complex syntax, (b) The variability in sentence length, and (c) The free order of the words within the sentences (unlike other languages where the sentences have almost a fixed word order) [2].

All of these complexities of Arabic grammar along with other language requirements such as grammar agreement rules make it necessary to develop grammar-aware applications that facilitate the understanding of this rich language, reduce the number of grammatical errors encountered by most of the average users, and handle the confusion introduced by the need to memorize the numerous grammatical rules.

Among the many challenges that NLP systems face during Arabic language processing, the challenge that is introduced by the Arabic Numerals that permeate text and speech is considered a significant one. The task of identifying Arabic numerals is challenging due to the following reasons [3]: (a) the same numeral can be represented by several expressions (b) each numeral can be represented in multiple formats: digits only (100), a multi-word sequence (hundred and fifty) and a mix of both (100 thousands) (c) a strong relationship exists between numerals and the counted objects, and many agreement rules may come into play involving gender, number, and case.

In the literature, many approaches and software applications have been proposed to perform syntactic-based and semantic-based processing [4, 5]. Word processing software is a good example of the applications that have proved their efficiency in the context of grammar checking and error correction. However, due to the unique and complex nature of the Arabic language, many syntactic and semantic errors may not be detected and corrected by word processing applications. In addition, the task of identification of numerals is not supported by such applications.

In this work, we present an approach that is dedicated to handling the challenge associated with Arabic numerals and their identification according to the standard Arabic numeral rules. The identification task is achieved so as to
check for violations of the gender agreement feature between the numeral and its counted object.

The rest of this paper is organized as follows: Section 2 presents the state-of-the-art research related to our work. In Section 3 a comprehensive discussion of the Numeral Rules and the proposed methodology is provided. Section 4 describes the experimental results. Finally, future directions and conclusions are provided in Section 5.

2. Related Work

Previous approaches [6–9] have focused on the context-independent identification of numerals, which converts from text format into digit format and vice versa; context-independent conversion has been applied to other languages including: Swedish [6], and Finnish [7].

Bengt Sigurd [6] presented an approach for identifying the relationships between the representation of the decimal system and Swedish numerals. In particular, this approach has focused on problems associated with the structure of numerals in natural languages such as English, German, French, Danish, Burmese, Hausa, Urdu and Finnish. Finnish was the major concern of this study. The interest was focused on developing algorithms that are able to automatically convert strings of digits that are interspersed with commas and decimal points as a sequence of words and vice versa. However, in this approach, the technical problems that arise when conversion rules are to be implemented on computers were not handled.

Karttunen [7] provided a complete description of the Finnish numeral system that formalizes other previous work taking into consideration the semantics of complex numerals. The Finnish numeral system is recognized using a single Finite State Transducer that maps numerals to the corresponding numbers with tags that indicate morphological features, ordinals and case.

All of these approaches [6–9] were proposed to perform numeral identification and conversion in an out-of-context manner. In addition, such approaches do not handle even small variations in the input number format; this is due to the fact that the authors of these approaches made a point of keeping their grammar small.

Bringert [10] proposed a Numerical Translator, which is a demonstration applet that uses a Grammatical Function (GF) Interpreter and a grammar for numerals to translate numbers among over 80 languages. However, as in the case of previous approaches, Bringert’s Numeral Translator cannot handle
all number variations. For instance, it can translate thirty four thousand one hundred into (34100), but cannot translate thirty-four thousand one hundred (with hyphen).

NUMEX is another approach proposed by [11] to identify monetary expressions and percentages in natural languages. However, the identification is limited to monetary expressions only and does not include all formats of digits.

Habash and Roth [3] presented an approach that performs the numeral identification task in a natural context. They proposed and validated a language-independent semi-automatic approach to build an application for evaluating number identification systems by exploiting hand-aligned parallel data. They also described a simple rule-based approach to Arabic numeral identification and evaluated it extensively.

For the purpose of presenting a standard grammar for reading Arabic numbers in both classical and modern formats, Al Anzi [8] presented two grammars for reading numbers of classical and modern Arabic language. The grammars make use of the structured Arabic counting system to present an accurate and compact grammar that can be easily implemented in two platforms. The first platform is used to convert from sentential representation to numerical representation, while the second platform is used to convert from numerical representation to sentential representation.

3. Methodology

In this research, we used Arabic Numeral Agreement Rules. These rules are the “Arabic Numeral Agreement Rules” — قواعد العدد والمعدود.

The Arabic language distinguishes between nouns based on their gender. The gender is usually referred to as: Masculine or Feminine. Using numbers to quantify these classes (or categories) of nouns is complex. The situation of counting a feminine (or a masculine) object in the Arabic language is more complex than in any other language, because the numerals are normally combined with the counted objects.

The Arabic language is extremely complex, there is a distinction between the single numbers or ordinal numbers “العربية المفردة” (these are the numbers from 1 to 10), the compound numbers “العربية المركبة”, (the numbers from 11 to 19), the decade numbers, which are referred to as “الفاتحة العقود” (numbers 20, 30, ..., 90) and the hundreds, thousands, millions, etc., numbers “العربية من sàn و مليون”.

The following subsection describes the distinct rules for numerals that motivate us to develop an application that considers them all.
3.1. Numeral rules

The general Arabic rules that combine these different categories of numbers along with the counted objects are the following [12]:

1. The numerals 1 and 2 are adjectives, so they follow the counted object and agree with its gender. For example, rajulun wāhidun (رجل واحد) ‘One man’, rajulani iṯnānī (رجلان إثنان) ‘Two men’. It is important to mention that in the Arabic language, adjectives follow the noun and agree with the preceding noun in gender, number, and case.

2. Numerals 3–10 have a unique rule of agreement known as polarity. In the rule of polarity, a feminine counted object agrees with a numeral in masculine gender and vice versa. For example, in the sentence: ṭalāṭu fatayātīn (ثلاثة بنات) ‘three girls’, the counted object fatayātīn ‘girls’ is feminine and the numeral ṭalāṭu ‘three’ is masculine. Whereas in the sentence: ṭalāṭatu rijaλiν (ثلاثة أشهر) ‘three men’, the counted object rijaλiν ‘men’ is masculine and the numeral ṭalāṭatu ‘three’ is feminine.

The Arabic numerals from 1 to 10 are written as follows [12]: 1 wāhid(tun) ( واحد), 2 ’ṯiṭnān(i) (إثنان), 3 ṭalāṭa(tun) (ثلاثة), 4 ’arba’a(tun) (أربعة), 5 ḥamsa(tun) (خمسة), 6 sitta(tun) (ستة), 7 sab’a(tun) (سبعة), 8 ṭamāniya(tun) (ثمانية), 9 tis’a(tun) (تسعة), 10 ’ašara(tun) (عشرة).

3. Numerals 11–19 are indeclinable, eternally in the indefinite accusative. Numbers 11 and 12 show gender agreement in the first digit, and 13–19 show polarity in the first digit.

Examples of the rule for numerals 11 and 12 are the sentences: aḥadu ’ašara rajulān (أحد عشر رجلاً) ‘eleven men’, iṯnā ’ašara rajulān (إثنا عشر رجلاً) ‘twelve men’, ’ṯidq ’ašarata imrātān (إثني عشرة أشيرة) ‘eleven women’, iṯnata ’ašarata imrātān (إثنتي عشرة أشيرة) ‘twelve women’. In these examples, the first digit shows gender agreement with the counted objects. The polarity between the counted objects and the first digit in numbers between 13 and 19 is shown in the following sentences: ṭalāṭatu ’ašara rajulān (ثلاثة عشرة رجلاً) ‘thirteen men’, ṭalāṭu ’ašarata imrātān (ثلاثة عشرة أشيرة) ‘thirteen women’, where the first digit shows gender disagreement with the counted objects (أملأ رجلاً). It is worthwhile to note that the gender of “عشر” ‘ten’ in numbers 11–19 agrees with the counted object ( unlike the standalone numeral 10, which shows polarity).

4. The numerals 20–99 are followed by a noun in the indefinite accusative singular as well. There is agreement in gender with the numerals 1 and 2, for example iṯnāni wa-išrūn rajulān (إثنان وعشرون رجلاً) ‘twenty two men’,
and polarity for numerals 3–9, for example َتَلَاثُ الطَّانُ وَعِشْرَتَنِاءُ رجلاَ (رَجُلًا) ‘twenty three men’.

5. Whole hundreds, thousands, etc. appear as the first terms of genitive constructions.

6. The rules for numerals 1 and 2, 3–10 and 11–19 are also applied when these numerals are combined with hundreds, thousands, etc. For example, in the numerals 101 and 102, the first digit shows gender agreement with the counted object, for example, ِيُعْتَنِ وَاً َتُقَادَتِ اَمْرََاتِ (رَجُلَا) ‘one hundred and one women’, while in the numerals 103–110, the first digit shows polarity with the counted object, for example, ِيُعْتَنِ وَاُتَعْمَانِيِ نَا رجلاَ (رَجُلَا) ‘one hundred and eight men’. Furthermore, in numerals between 111 and 119, the rules of agreement and polarity mentioned in rule 3 apply exactly here, for example, ِيُعْتَنِ وَاً تَلَاثُ الطَّانُ ‘one hundred and thirteen men’.

In this research, we developed an approach that checks on the level of compliance with the previously mentioned rules for numerals and ensures that these rules are handled properly and satisfy the gender agreement feature.

3.2. The proposed approach

In this work, we present an approach with the name “Arabic Numerals Checker” that attempts to handle the previously mentioned numeral rules.

Our present work is only concerned with identifying any given number into the corresponding Arabic numeral without paying attention to preceding steps such as parsing that return, as their output, the numerals and their counted objects. Therefore, this work can be considered as an independent subroutine that can be integrated with other routines to form a larger, neat system.

using a separate entry in the vocabulary, then the lexicon will be too large, which adds more complexity.

To solve this problem, we process the numerals into a set of basic morphemes, such that each numeral consists of a single morpheme or a morpheme followed by some affix. For instance, “١٢٣” consists of the morpheme “١٢٣” and the affix “١٢٣” in turn, consists of the morpheme “١٢٣” and the affix “١٢٣”. Thus, we only need to store two entries in the lexicon for these numerals, namely, “١٢٣” and “١٢٣” rather than dedicating four separate entries for them.

This method of preprocessing allows us to reduce the size of the lexicon so that it contains the basic morphemes only, and thus, facilitates the task of numeral identification. To model this approach, we used Finite State Transducers (FSTs), which precisely represent the lexicon and the transformation between morphemes to represent the sequence of morphemes. As an illustrative example, Figure 1 provides the FST for the number “Four” - “١٢٣” and all its possible forms.

At this point, it is important to illustrate that in our implementation of the “Arabic Numerals Checker” approach, we store each basic numeral as a morpheme. That is, the set of morphemes includes: “١٢٣”, “١٢٣”, “١٢٣”, “١٢٣”, “١٢٣”, “١٢٣”, “١٢٣”, “١٢٣”. These morphemes are stored in a simple two-dimensional

Figure 1. FST for the forms of “١٢٣”.
array in which each morpheme is stored in a place that corresponds to its numeric value. The integers from 3 to 10 correspond to these numeric values. For instance, the numeral “Four” - “اربع”, is stored in the index [4, 0], the numeral “Five” - “خمس” in the index [5, 0] and so on. This way of organizing and storing numerals facilitates the very fast and efficient access to the needed numeral.

3.3. System functionality

As discussed above, the index-based retrieval of a particular numeral as well as the decrease in lexicon size of our approach (which includes the basic numerals only) reduces the number of operations that must be performed in order to identify the number and transform it into the corresponding multi-word sequence numeral. Figure 2 briefly represents the main processes that are used to complete the identification and transformation tasks and the flow of data between them.

Our implementation of the “Arabic Numeral Checker” covers all numbers in the range 1–999,999,999,999. The transformation of any large number

![Diagram](https://via.placeholder.com/150)

**Figure 2.** Level-1 data flow diagram.
into its corresponding numeral was accomplished efficiently by implementing only a few functions. In fact, we exploit the patterns in which numerals are composed to form larger numerals. For example, the numeral 987,654,321 can be viewed as being consisting of three blocks of digits, and each block consists of the same components (hundreds, tens, and ones), therefore, the identification process that is applied on the first block to identify the first three digits can be repetitively applied on the second and the third blocks to identify the remaining digits. In our approach, the number to be transformed is divided into blocks of three digits each. Then each block is passed through the same sequence of functions in order to be written in the linguistic (numeral) format. Thereafter, the recognition of (thousands) (الآلف) (millions) (مليون) and (billions) (مليارات) takes place as needed depending on the number of blocks and their locations.

As shown in Figures 3 and 4, the number to be transformed into numeral format is divided into blocks consisting of <hundreds, tens, ones>, each block is processed by the same functions, hence the amount of work needed for large numbers is reduced significantly. After each block is processed, the word “ألف” is attached with the second block, the word “مليون” “الثامن” with the third block, and the word “مليار” “الثامن” is attached with the fourth block.

Figure 3. Structure tree for numbers.
4. Experimental Results

Listed in Figures 5–10 the five scenarios for our approach. Each figure represents a different rule and illustrates the result via a pictorial snapshot. In addition, Figure 11 represents a piece of running text from a newspaper, which includes many numbers in the digit format.

Figure 5 illustrates Rule 1 where numerals 1 and 2 show gender agreement with the counted object. The figure shows that when the number 2 is inserted as an input and the gender of the counted object is selected to be masculine (man), the output result is ‘Two men’. It is important to illustrate here that in Figure 5 and all other figures, we have used “Man” and “Woman” to refer to masculine and feminine counted objects, respectively.

Figure 6 is designed to illustrate Rule 2, which states that the numerals between 3 and 10 show polarity with the counted object. As shown clearly, when the number 9 is input and the counted noun is feminine, the result is ‘Nine women’ which is in the masculine format.

Figure 7 shows that the numeral 11 has gender agreement in the ones (i.e. in the first digit) as stated in Rule 3. When the gender of the counted object is feminine, both digits of the numeral agree with this gender and they are written in the feminine format ‘Eleven women’.
Figure 5. Example of Rule 1. Gender agreement is indicated between the numeral (اثنان) ‘Two’, and the masculine counted object.

Figure 6. Example of Rule 2. Gender polarity is indicated between the numeral (تسعم) ‘Nine’, and the feminine counted object.

Figure 7. Example of Rule 3. Gender agreement is indicated between the numeral (عشرة) ‘Eleven, and the feminine counted object.

Figure 8. Example of Rules 4 and 5. Figure 8 illustrates the identification of numerals that combine Rules 4 and 5, where the inserted input is 783 and the gender of the counted object is masculine, the output (سبع مئة وثلاثة وثمانون) ‘Seven hundred and eighty
three men' - shows that the word for hundred appears as the first term of genitive constructions (سَمَعَ مَنَةً), and that there is polarity in gender with the numeral 3.

Gender agreement is indicated between the hundreds digit and the masculine counted object, and polarity in gender with the numeral (ثلاثة) ‘Three’ appears in the first digit.

Figure 9 shows a comprehensive example that presents the ability of our approach to correctly identify large numbers and transform them into their corresponding numerals. The input number uses Rules 4, 5 and 6 and the output numeral shows the effects of these rules, where the gender is masculine. The English translation of the output numeral is ‘Two hundred and thirty four billion, four hundred and fifty six million, seven hundred and eighty nine thousand and four hundred and eleven’. Gender agreement is indicated between the first digit (أَفْدَعُ عَشَرُ) ‘Eleven’ and the masculine counted object, and polarity in gender with the numerals (34, 56, and 89).

Figure 10 presents the intelligent feature of our scheme, which is the explanation feature that provides the user with a logical explanation of the output result. The figure shows an explanation of the result of identifying the numeral that is inserted in Figure 9, and shows that the inserted number satisfies Rules 2, 3, 4, 5 and 6. Figure 11 represents a piece of running text from a newspaper, which includes many numbers in the digit format.
5. Evaluation and Comparative Results

In most NLP systems, the evaluation approaches are divided into two categories, the black box, in which the authors test their approach by evaluating only the input and the output of the system under evaluation. The other category is the white box evaluation, in which the authors evaluate the functionality of the various parts of the system and assess how the sub-parts are working as well [13].

Conducting a white box evaluation and comparison of our results with other related approaches like that of [3] was not feasible due to the fact that their working is different than ours and that we do not have access to the various workings of the system and thus cannot evaluate each sub-part of the system and analyze the system errors according to the technique of error analysis that we have followed. Therefore, we chose to compare our Arabic Numerals Checker with commercially available programs that perform the task of automatic numeral identification.

In our work, we have followed the black-box evaluation approach to assess the output of our system and that of the commercial systems in correspondence with a particular input and analyze the type of errors that both systems fell in.

A set of 100 Arabic sentences was used to compare the results of Arabic Numerals Checker with those obtained by numeral translation programs available on the market and to evaluate their identification accuracy. This set was obtained from different Arabic newspapers. The set of sentences includes
numbers and counted objects of feminine and masculine genders. The sentences are picked to be short and simple so as to be easier for the reader to understand.

A set of 100 Arabic sentences was used to compare the results of Arabic Numerals Checker with those obtained by commercial software packages available on the market [14, 15] and to evaluate their identification accuracy. This set of sentences was obtained from different Arabic newspapers. The set of sentences includes numbers and counted objects of feminine and masculine genders. The sentences are picked to be short and simple so as to be easier for the reader to understand.

A summary of the evaluation results is shown in Tables 1 and 2. We evaluate the correctness of the results using three levels as: fully-correct, partially-correct and wrong. The output is considered as full-correct if the Arabic Numerals Checker gives a fully correct identification of numbers and respects the gender agreement rules. The output is considered as part-correct (partially-correct) if the number is correctly converted to its corresponding numeral but there is some inconsistencies regarding the gender agreement rules or the standard Arabic language grammar or spelling. The output is considered as wrong if the Arabic Numerals Checker wrongly identifies a number or did not identify a number at all.

Of the 100 Arabic sentences, our approach resulted in 93 fully-correct sentences and 7 partially-correct, while there were no wrong sentences at all. Using the commercial programs [14, 15], however, The average of the experimental results shows that there were 84 full-correct sentences, 9 part-correct and 7 wrong sentences. The overall correctness is shown in the bottom row, which indicates the percentage of the input sentences marked as fully-correct, partially-correct, or wrong, in total.

It is worthwhile to mention here that even though we got 7% of the sentences partially correct (part-correct), the results are still excellent because having a partially correct sentence is not a serious problem, since the number is correctly transformed into its corresponding numeral, but there is inconsistency regarding Arabic grammar. An example of a part-correct numeral is “ثمانية ألفاً ومئة وثلاثة وعشرون”, where “ثمانية” is in the singular form, and it should be “ثمانية” in the plural form in order to be consistent with “الإف” which is plural.

Table 1. Correctness of Arabic Numerals Checker.

<table>
<thead>
<tr>
<th>Sentences</th>
<th>Fully-correct</th>
<th>Partially-correct</th>
<th>Wrong</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>Commercial Software</td>
<td>84</td>
<td>9</td>
<td>7</td>
<td>100</td>
</tr>
<tr>
<td>Total (percentage)</td>
<td>84%</td>
<td>9%</td>
<td>7%</td>
<td>100%</td>
</tr>
</tbody>
</table>
6. Conclusions and Future Work

The Arabic language is a rich language in terms of its syntax, semantics and morphology. Among many language rules that should be considered during the processing of the Arabic language, “the rules for numerals and their counted objects” are of special concern. In this paper we have present an efficient, explanation-based approach that transforms a particular number from a string of digits into a multi-word format taking into consideration the gender of the counted object.

The system is based on a deep morphological analysis using Finite State Transducers (FSTs) to represent numerals and the transformation process. The use of FSTs reduces the size of the lexicon significantly, yet provides promising and accurate results. To facilitate the fast access and retrieval of morphemes in the FST, an index-based technique for storing morphemes is applied.

The process of identifying numerals is fundamentally based on viewing the number as consisting of several blocks, each with the same constituents. This perspective reduces the amount of work needed to transform numbers (esp. large ones) into their corresponding numeral format.

As future work, our application will be extended to include parsing techniques that extract and return the number (in the form of a digit string) along with its counted object from a natural language document, then apply the numeral checking functionalities of our present approach to transform it into a string of words. Another direction will be to integrate our approach with other text editor applications so that the task of identifying numerals and transforming them into strings of words can be embedded as a vital part of the grammar checking task.

References


[14] Excel
http://www.infotechaccountants.com/forums/showthread.php/2109--
-excel. (last accessed 30, December 2009).

30, December 2009).