Lang Resources & Evaluation 46 (2012) 4, pp. 543–563

DOI: 10.1007/s10579-011-9146-z

A real time Named Entity Recognition system for Arabic text mining

Harith Al-Jumaily · Paloma Martínez · José L. Martínez-Fernández · Erik Van der Goot

Abstract Arabic is the most widely spoken language in the Arab World. Most people of the Islamic World understand the Classic Arabic language because it is the language of the Qur'an. Despite the fact that in the last decade the number of Arabic Internet users (Middle East and North and East of Africa) has increased considerably, systems to analyze Arabic digital resources automatically are not as easily available as they are for English. Therefore, in this work, an attempt is made to build a real time Named Entity Recognition system that can be used in web applications to detect the appearance of specific named entities and events in news written in Arabic. Arabic is a highly inflectional language, thus we will try to minimize the impact of Arabic affixes on the quality of the pattern recognition model applied to identify named entities. These patterns are built up by processing and integrating different gazetteers, from DBPedia (http://dbpedia.org/About, 2009) to GATE (A general architecture for text engineering, 2009) and ANERGazet (http://users.dsic.upv.es/grupos/nle/?file=kop4.php).

H. Al-Jumaily (✉) · P. Martínez
Computer Science Department, Carlos III University of Madrid, Av. Universidad 30,
28911 Leganés, Madrid, Spain
e-mail: haljumai@inf.uc3m.es

P. Martínez
e-mail: pmf@inf.uc3m.es

J. L. Martínez-Fernández
DAEDALUS – Data, Decisions and Language S.A., Avda. de la Albufera,
321, 28031 Madrid, Spain
e-mail: jmartinez@daedalus.es

E. Van der Goot
EC Joint Research Centre, Via E. Fermi, 27549 Ispra, Italy
e-mail: erik.van-der-goot@jrc.it

Keywords Arabic language · Text mining · Named Entity Recognition · Event detection · Morphological analysis · Root extraction

1 Introduction

We currently have many opportunities to obtain a wide variety of information from the Internet. This information is growing at an exponential rate. Most of it is written in natural language because the Web was designed for human reading and understanding, not for machine recognition and interpretation. The huge volumes of information and the widely extended use of Internet have attracted researchers to face up to these new challenges and to improve Internet services. Information Retrieval (IR) is one of the most important services that provides tools to achieve relevant and complicated searches on Internet. Therefore, there is fierce competition between the gigantic search engines such as Google, Yahoo, etc., to provide the best application with more performance and accurate information.

Text mining technology is becoming one of the major issues in IR. It highlights the relevant information in the text that can be used for different applications providing the users with powerful search functionality that goes far beyond text search (Steinberger et al. 2008). So, text mining applying techniques can help to confront the challenge of searching through huge volumes of information (Martin and Van der Goot 2009). For example, the web applications which are used to detect news about people, organizations and places, as well as those which may give earlier warnings about medical and health-related topics or natural disasters, all require more work and efforts to be made in text mining and pattern recognition (Best et al. 2007). The framework in which the research work described in this paper has been developed includes a real time monitoring system for news channels in order to detect the appearance of specific words or expressions, which have been collected as a result of previous works. A large quantity of news must be processed in a short period of time so efficient methods to look for these words in the news are needed. Names of people, places and organizations must also be tagged so a Named Entity Recognition (NER) process is required. Thus, our system has been designed to recognize two types of elements in the Arabic language: named entities and a set of common words (nouns and verbs). As regards NER, in this paper we will consider only a basic categorization scheme for named entities made up of three main classes: Person, Location and Organization. As regards the common words, we apply the recognition task to detect nouns and verbs. Khoja's Stemmer (Khoja and Garside 1999) has also been integrated to extract the linguistic roots of the detected nouns and verbs. Our approach tries to minimize the impact of Arabic prefixes and suffixes on the quality of the recognition patterns, which is considered one of the major problems found in Arabic (Afify et al. 2006). A part-of-speech process has not been considered because it would take too much time to process each piece of news so on-the-fly monitoring would not be feasible.

The rest of this work is organized as follows: In Sect. 2, we provide a brief introduction on the Arabic language to make it easier for a non-native Arabic reader to understand the difficulties of the Arabic language. In Sect. 3, some related works

are presented and discussed. Section 4 describes the architecture of the system. Section 5, shows defined experiments and discusses the results. In Sect. 6, some conclusions and future works are presented.

2 Why the Arabic language?

Despite the number of Arabic Internet users having increased considerably in the last decade (Fig. 1) (Internet World Stats, Usage and Population Statistics, http://www.internetworldstats.com/), the content of Arabic digital resources on the Internet is still lower than its actual weight as a language. These resources are less than 1% of all Internet content, while people who speak Arabic natively represent 5% of the world's population (El potencial de la Red en a´rabe, 2010), so a significant increase in Arabic content is expected and tools, like those proposed in this work, are needed to process this content automatically.

The Arabic language is one of the Semitic languages that is used by people living in the Middle East, and North and East of Africa. It is the official language throughout the Arab World, which consists of 25 countries although each one of them has its own regional variants. In addition, many people of the Islamic World (non-Arab countries such as Turkey, Iran, Pakistan, etc.) understand Classic Arabic (CA) because it is the Language of the Qur'an. Nevertheless, most writing in the Arabic World such as books, newspapers, magazines, official documents, etc., is in Modern Standard Arabic (MSA), which is one of the official languages of the United Nations. "The major difference between the MSA of today and the CA of yesteryear is that the former is truly a living language subject to the many influences of the Arabic spoken dialects, whereas the latter is a frozen, static entity" (Kaye 1991). For this reason, we have only considered MSA in this work.

The Arabic alphabet consists of 28 letters; three of them are used as long vowels. Most of these letters have different shapes depending on whether it will be connected at the beginning, middle or end of the word. The Arabic alphabet also contains three short vowels that are normally placed above or below the corresponding letter. On the other hand, the reader should know the difference between the basic concepts of Arabic; root, stem, and vowel patterns. Arabic text is written and read from right to left. Most of the words are built from roots, except the common names and particles. At the same time, these words can be analyzed to obtain the base roots. Around 64% of the Arabic roots are made up of three letters (Khoja et al. 2001). A root can also be formed from two or four letters. In limited cases, a root can have more than four letters.

The Arabic language has highly inflectional and derivational difficulties because there are many irregular words (Al-Zoghby et al. 2007), and there are many vowel patterns, each defining a grammatical state of the stems. For example, let us consider the root (كتبktb[1]) '*write*' which contains three letters (كk + تt + بb), to derive all the

[1] In this paper the Arabic words are presented using the HSB transliteration schema (Habash et al. 2007) as follows: ء', آĀ, أÂ, ؤŵ, إĂ, ئŷ, اA, بb, ةh, تt, ثθ, جj, حH, خx, دd, ذð, رr, زz, سs, شš, صS, ضD, طT, ظĎ, عς, غγ, فf, قq, كk, لl, مm, نn, هh, وw, ىý, يy,á,u̯,i̯,ã́,ṹ,ĩ́,~ , . , - _,.

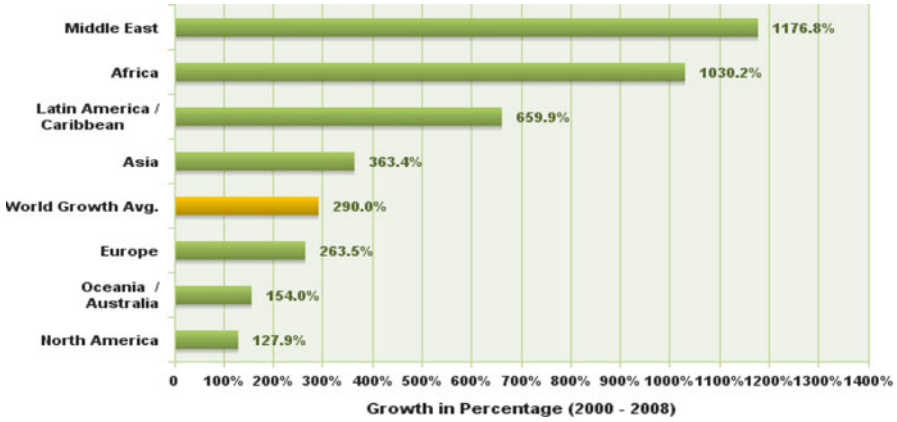


Fig. 1 World Internet growth between 2000 and 2008

possible stems from this root we should replace the letters (- ف+-ع- ي+ ل) of all the vowel patterns with the three letters (ك+k ت+t ب+b) of the root respectively, as shown in Table 1. So far, it seems easy: all that we need to do is apply the vowel patterns to the roots to obtain almost 70% of the Arabic stems. Nevertheless, the problem here is that there are more than 5,000 Arabic roots and more than 1,500 vowel patterns (Alqatta Alsaqly 1999). In addition, many of these forms consist of the same set of letters differing only in the way the short vowels are used. For example, in Table 1, the stem (كَتَبَ*kat~aba*) is the past tense of the verb but if this stem appears with other set of short vowels as (كُتبْ*kutb.*) it is a noun meaning '*books*', and the same situation can be observed between (كُتاب*kutAb*) and (كتاب *kitAb*). Here, the problem can be aggravated because the short vowels are typically not written in normal texts such as (books, newspapers, magazines, official documents, etc.). Sometimes, they are written when word ambiguity cannot be solved from the text. So the reader must be familiar with the language to understand the missing vowels.

Another problem is *al-hamzah* (ء ') which is an additional letter in the Arabic alphabet, representing the sound of a glottal stop. It can be written in different forms; alone (ء) at the end of a word, above or under the letters (ا *A*), (و *w*), and (ى *ý*), such as (أ *Â*), (إ *Ă*), (ؤ *ŵ*), and (ئ *ŷ*). This often leads to orthographical confusion, thus people do not usually write the *al-hamzah* in the right place or it can be totally ignored. For example, if we search the words (إسبانيا *ĂsbAnyA*), (أسبانيا *ÂsbAnyA*), and (اسبانيا *ÂsbAnyA*) in Google which are different ways of writing '*Spain*' in Arabic

Table 1 An example showing roots, stems, and vowel patterns

Vowel patterns	Stems	Grammatical state	Meaning
فَعَلَ *Faç~ãla*	كَتَبَ *kat~aba*	Past verb	He has written
فُعلْ *fuçl*	كُتبْ *kutb*	Noun	Books
فِعال *fuç*Al	كُتاب *kutAb*	Noun	Writers
فِعال *fiç*Al	كِتاب *kitAb*	Noun	A Book

using different types of the *al-hamzah,* we will get approximately the same number of pages (4.7 million). Although the second spelling of the previous example is not a variety but a wrong spelling of Spain in Arabic, Google considers the three letters (*Ǎ*), (*Â*), and (*A*) as the same letter to resolve orthographical confusion. In addition to the *al-hamzah*, there are other letters that can be written in different shapes in Arabic, such as using *tatweel* (-) to elongate the Arabic letter, and using the modified letters such as (پp, ڤv, چ گ ڤg, etc.) to write foreign words (Habash 2010).

Finally, as we said, one of the major problems of the Arabic language is the prefixes and suffixes because there is a large number of them, approximately more than 80 prefixes and 200 suffixes. A prefix length can range from 1 to 4 letters, while a suffix length can range from 1 to 6. For example, if we use the suffix (انAn) with the noun stem (كتابktAb) '*book*' we will obtain a new word (كتابانktAbAn) '*two books*'. In addition to the large number of them, the compatibility problem must also be resolved between them and with stems (Buckwalter 2004). In this work, the impact of using the Arabic affixes in our system will be studied and a verification algorithm will be implemented to improve its performance.

3 Related work

In this section we study and analyze some of the important efforts on Morphological Analysis (MA) and NER for Arabic. We start by presenting the Buckwalter Arabic Morphological Analyzer (BAMA) (Buckwalter 2004) which is considered a pioneering work for the Arabic language. It consists of a dictionary of lexicons of Arabic stems, prefixes, and suffixes, with truth tables to indicate a correct combination of these affixes. It provides morphological categories such as Function word, Nouns, and Verbs. It uses Buckwalter transliteration, which can be converted directly to Unicode Arabic with a minimal amount of automatic processing. Many important Arabic researches are based on BAMA such as; MADA + TOKAN (Habash et al. 2009) which is a freely available toolkit for Arabic NLP applications. MADA handles the Morphological Analysis and Disambiguation for each word, and TOKAN presents the results in a wide variety of customizable formats. SAMA 3.1 (Maamouri et al. 2009) is another Arabic morphological analyzer which is based on BAMA. SAMA is considered the continuity of the previous BAMA releases.

It is worth highlighting other efforts such as, AMIRA (Diab 2009) which is an online toolkit for Arabic tokenization, lemmatization, POS-tagging and Base Phrase chunking. It has been widely used for different NLP applications due to its speed and high performance. Serf (The Arab League Educational, Cultural and Scientific Organization (Alecso) 2007) is an Arabic Morphology analyzer that uses the trilateral and quadrilateral roots as input, the output is a set of all the grammatical states of these roots. The main limitation of Serf is that the inputs are only roots. NOOJ (Silberztein 2002) is a developed NLP environment in many languages. It offers a free plug-into provide morpho-syntactic tools for Arabic processing. Khoja's Stemmer (Khoja and Garside 1999) is also a valuable system for the Arabic language. It uses an algorithm to remove the prefixes and suffixes from the input words and then it extracts roots from these words. In an earlier phase, it removes

stop words and strange words from the original text. A stop word is a commonly used Arabic word. A strange word is a foreign word written using Arabic script. Although removing the affixes is not sufficient for many NLP applications (Al-Sughaiyer and Al-Kharashi 2004), we believe that root extraction can help in the ED task because most of the Arabic words can change their orthographic forms when their grammatical states are changed.

On the other hand, NER is considered to be an important task since it helps to improve the performance of many NLP applications (Benajiba et al. 2008). It is a valuable source to capture the semantic content of a written text. However, very little research into NER for Arabic has been published (Benajiba 2009) due to the lack of the related resources and the limited progress made in Arabic NLP in general (Shaalan and Raza 2008). Many techniques are used to build NER systems for Arabic. For example, Abuleil (2004) used a set of rules to predict where the Arabic proper names are located in the text. It states that entity names seem to appear close to trigger words (keywords or special verbs). So it assumes that these names are found in a space of 10 words to the left and 10 words to the right of the trigger word. ANERsys which is a NER system built exclusively for Arabic texts based-on n-grams and maximum entropy approach is presented in Benajiba et al. (2007). They developed their own corpora and gazetteers to train, evaluate and boost the system. They obtained an improvement with respect to a baseline results without using any POS-tag information or text segmentation. In Benajiba et al. (2010) the authors achieve a significant, high-performance Arabic NER system by using lexical, syntactic and morphological features. In Shaalan and Raza (2008) a system for Arabic NER is presented which consists of two main processing resources: a dictionary of names (whitelist or gazetteer), and a grammar, in the form of regular rules to recognize the Named entities. A filtration mechanism is used as a blacklist to reject matches returned by rules but which are invalid entities. The system is evaluated using its own corpora which have been tagged in a semi-automated way. GATE (2009) is a Java suite of tools, which is widely used in NLP tasks, including information extraction in many languages. It has an Arabic module as one of the plug-ins in the CREOLE directories. GATE can be used to extract basic entities, such as date, name, location, organization, etc. The main problem of the tools such as GATE is that they were developed mainly to analyze non-Arabic language, however, plugs-in have been added in them to make sense of the Arabic (Farghaly and Shaalan 2009).

From the aforementioned overall trends the NLP for Arabic is directed towards the Morphological Analysis (including root extraction), and NER. We agree with (Benajiba et al. 2009) that from the NER system's point of view, the recognition task in Arabic is relatively different from performing the task in English due to the aforementioned problems and because the Arabic script lacks capital letters. Thus we believe that performing text mining on the Arabic language requires more effort and poses special challenges (Halpern 2007). Although our system is directly related to NER, we are conscious that the morphological categories (nouns and verbs) are vital to detect events and provide earlier warnings about them. So, we believe that a system is necessary that provides information such as nouns and verbs in addition to the common Named entities. The root extraction is also included in our system to make the identification of these words easier.

Finally, in this work we are interested in studying the impact of the Arabic affixes on the performance of our system. In other words, we will try to reduce the impact of these affixes and to improve the recognition results. If an Arabic token (prefix-stem-suffix) is recognized then a verification process is used to ensure that the three combination prefix-stem, stem-suffix, and prefix-suffix are compatible. A stem may be one of the Named entities or the morphological categories considered in this work. Finally, our system has been designed to detect the Named entities which can be represented by more than one word. It can be easily integrated with the rest of the system, providing output results in XML format and UTF-8 codepage to simplify subsequent exploitation.

4 System architecture

In this section, we will explain our proposed system architecture for Arabic text mining and analysis (Fig. 2) in more detail. When we started to design our system architecture, we realized that this architecture must meet the following requirements:

- Use the UTF-8 codepage for the I/O.
- Use existing systems of NLP with some adaptation to reduce the level of effort required to develop our system.
- Allow the user to append information into the system dictionary ΣPMG; this dictionary is divided into three parts: ΣP for the prefixes, ΣM for the morphological categories, and the gazetteer ΣG for the Named entities. Each one of these dictionaries is a text file that can be edited by any text editor and the

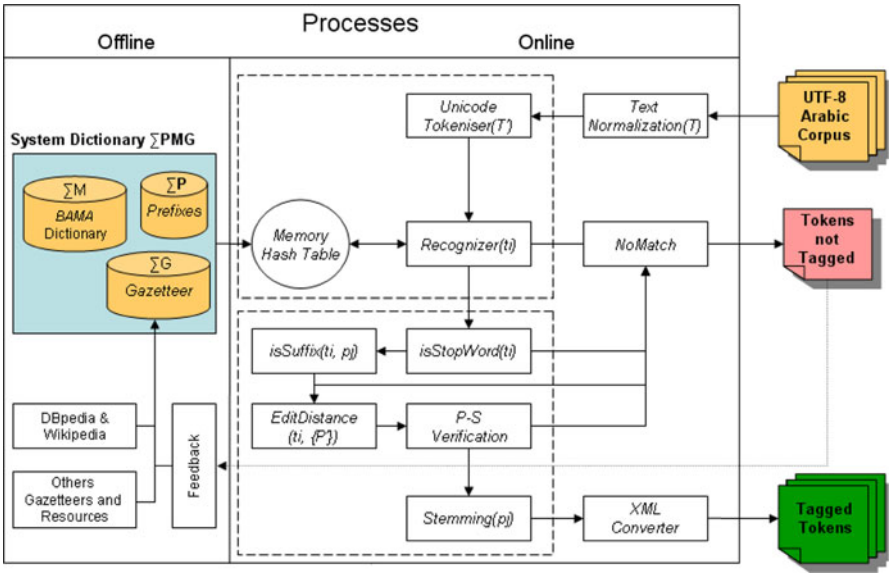


Fig. 2 A pattern-based architecture

user can manage the information of these dictionaries according to his/her requirements.

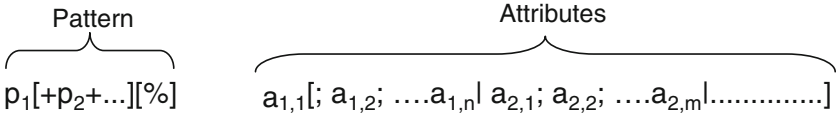
- The system provides automatic mechanisms to manage the prefixes and the suffixes of the Arabic language. These mechanisms help to optimize the number of words in the dictionary effectively. For example, suppose the number of prefixes and suffixes is (N and M) respectively, this means that the definition of one word in the dictionary allows us to create (1 + N + M + (N*M)) forms of that word in the online processing.
- The system has been implemented using Java, to facilitate the integration of the architecture components, it is easy for an inexpert user, and the system allows the addition of new modules according to future requirements.
- The output is in XML format to make the information exchange with other systems easier.
- The architecture could be adopted for analyzing other languages that are currently written using the Modified Arabic alphabet such as Kurdish, Persian, etc.

The architecture built to meet these requirements contains two types of processes: offline and online. In the following, a brief description is shown for each these processes:

4.1 System dictionary ΣPMG creation

The main objective of these processes is the creation of ΣPMG. This dictionary is used to save all patterns that were retrieved from various information resources. The objective of creating ΣPMG offline is to accelerate the system performance.

Currently ΣPMG contains about 94,000 patterns, each one of which is associated to its attributes. A pattern (P) is associated with a set of attributes (a_1| a_2|…) and each one of these attributes can have one or more values. A pattern specifies a reference word in the Arabic language while the attributes specify the information and the categories of the corresponding pattern. Almost all of these patterns are associated with the morphological categories while 13% of them are associated with the Named entities. The formal definition of a pattern and its attributes are shown below:



We can distinguish three parts in ΣPMG; Prefixes ΣP, Morphological Resources ΣM, and Gazetteer ΣG. In the following a brief description of each of the dictionaries is shown:

(a) Prefixes ΣP: The Arabic prefixes are stored in the Prefix part, where each prefix is associated with only one attribute which has (P) as a value. In this part, we define all the possible letters of the prefixes. In the following, we show the content of ΣP:

Pattern	a_1
ا	P
ب	P
ت	P
س	P
ف	P
ك	P
ن	P
ل	P
و	P
ي	P

(b) BAMA Dictionary ΣM: The second part of ΣPMG is the morphological dictionary ΣM that has been automatically generated from BAMA (Buckw-alter 2004). To create ΣM, the *dicstems* dictionary of BAMA was converted from the Microsoft Windows Codepage 1256 (Arab) to UTF-8 codepage. In addition, some adjustments to adapt the results according to the ΣPMG format were made. In the following, an example of a pattern in ΣM is detailed:

Pattern	a_1	$a_{2,1}$	$a_{2,2}$	a_3
ابحر%	أَبْحَر	set sail	travel by sea	PV

This example defines a pattern that shows the normalized pattern of the word (أبحر *ÂbHr*). The symbol (%) that is associated with the pattern indicates that this pattern allows the aggregation of suffixes, for example, the letter (ت *t*) in (ابحرت *AbHrt*) which means '*She/It travels*'. In the following, we explain each one of the pattern's attributes:

– (a_1) has one value (أَبْحَر *Âab.Har*) which indicates the truly complete form of the pattern, where all the short vowels are reviewed.
– (a_2) has two values ($a_{2,1}$ = *set sail* and $a_{2,2}$ = *travel by sea*) to indicate the translation of the word to English.
– (a_3) has one value (PV) to show the morphological category of the pattern. Our morphological categories agree with the morphological categories of BAMA (Buckwalter 2004). The three main categories are Function Word (particles, pronouns, and other words that do not function as Nouns or Verbs), Nouns, and Verbs (Perfect Verb, Imperfect Verb, and Imperative Verb).

(c) Gazetteer ΣG: The third part of ΣPMG is the gazetteer ΣG that aims to define patterns with the Named entities. Normally the patterns of this part of the dictionary are associated with only one attribute, as an example, the following shows some patterns.

Pattern	a_1
ثاباتيرو	Per
أمم+متحدة	Org

Although ΣG forms only 13% of ΣPMG, i.e. it is smaller than the morphological dictionary; the offline processes that were used to generate ΣG have been more complicated and more expensive because various digital resources have been used in different ways. In order to generate about 12,000 non-repeated patterns for ΣG, the estimation of the total time spent in this process was approximately one person for 420 h of work. This time has been averaged between providing a number of programs for automatic processing, using various tools and databases to save and manage the generated data, and finally manual work to adapt these data to the ΣPMG requirements.

In the following the main resources of data used in this work are explained:

- *DBpedia/Wikipedia:* DBpedia is a project aimed at extracting structured information from Wikipedia for IR (DBpedia 2009). DBpedia provides an automatic conversion of the existing information in Wikipedia to RDF (Resource Description Framework) (The World Wide Web Consortium (W3C) 2009). Through the DBpedia website, users can download the RDF databases (triple storage). A triple is a statement that contains subject, predicate and object. For example, we have located all the triples that have a predicate as Country, Capital, City, etc. and categorized them as a location in our system, and so on for the other categories. Although the Arabic version of the RDF is not included in DBpedia, we have used the English version and terms have been translated using Google. Manual revision was applied to check the translation.
- *Other Gazetteers and Resources:* From the Internet and other tools for entity recognition, we obtained prepared and classified lists of data according to common categories. Among the gazetteers that have been used are the Arabic Gazetteer lists of the GATE (2009). Each one of these lists contains a different category, for example, the *country.lst* list contains a set of the Arabic countries, and the *country_world.lst* list contains the set of the remaining world countries. Moreover, there are female names, male names, surnames, organizations, mountains, etc. Another gazetteer exploited is ANERGazet (Natural Language Engineering Lab, http://users.dsic.upv.es/grupos/nle/?file=kop4.php) which contains a collection of three Gazetteers, (i) Locations: a gazetteer containing the names of continents, countries, cities, etc., (ii) People: a gazetteer containing names of people manually collected from different Arabic websites, and finally (iii) Organizations: containing names of companies, football teams, etc.
- *Feedback:* It is an offline process which is used to maintain the lexicons of the system dictionary. This process is applied semi-automatically to detect the words which have not been recognized by the system. All words which are not recognized by our system are saved in the text file 'Tokens Not Tagged'. The content of this text file is analyzed according to the count of the appearance frequency of tokens. The tokens with a highest frequency count are removed from the analysis, and the rest of them are analyzed manually. As a future work we are going to incorporate full automatic algorithms to detect entities depending on trigger works and a list of matching rules (Abuleil 2004; Pouliquen and Steinberger 2007).

4.2 Implementation issues for the classification algorithm

The main objective of these processes is to apply the classification algorithm according to the proposed architecture. The algorithm starts when one or more UTF-8 Arabic documents is entered. The online processes are run according to the following sequence.

(a) Normalization (T): This process is run in the first place to convert the entered corpus (T) into a normalized corpus (T'). In this process the various types of *al*-hamzah (أ *Â*, إ *Ǎ*, and آ *Ā*) are unified to (ا *A*), the short vowels and the elongation letters are removed from the text. Sometimes the elongation letter (-) is used for text decoration, or for writing the short vowels upon it, such as (ـَـ *ã*).

(b) UnicodeTokenizar (T'): This process receives the normalized text (T') and returns a set of tokens ($\{t_1, t_2,\ldots,t_n\} \in T'$). Each token is a unigram word. The white space character is used to split tokens. Therefore, a token may be an Arabic word, English word, numbers, symbols and punctuation marks.

(c) Recognizer (t_i): For any token ($t_i \in T'$), the Recognizer process looks in the Memory Hash Table and returns a set of patterns $\{P_{1..m}\}$ which match t_i. $\{P_{1..m}\} = \{p_1, p_2,\ldots,p_m\}$ can be one pattern or more than one pattern. For example, if t_i is the Arabic token (وعدوانها *wçdwAnhA*) '*and its aggression*' then the Recognizer process will return a set of 6 patterns, as follows; $\{P_{1.6}\}$ = {وعد *wçd*, وعç *wç*, عدو *çdw*, عدوان *çdwAn*, عدوا *çdwA*, عد *çd*} which mean respectively '*promise*', '*make conscious*', '*run*', '*aggression*', '*infection*', and '*count*'. Each one of these patterns forms a part of the input token ($p_j \in t_i$). The UnicodeTokenizar and the Recognizer of our system is based on the EMM search engine of the JRC (2009).

(d) NoMatch (t_i): This process is run when {P} = {''}, i.e., the set of the matched patterns is empty; the NoMatch process writes t_i in the TokensNotTagged file. As we said, this file is periodically studied to detect relevant words not being recognized by the system.

(e) isStopWord(t_i): If the set of the matched patterns {P} is not empty this process is run to check whether t_i is a stop word. A list of more than 500 stop words has been created as a dictionary to discard the words that are frequently used. This list contains Prepositions, Adverbial particles, demonstratives, pronouns, etc.

(f) isSuffix(t_i, p_j): If t_i is not a stop word then we reduce the size of the set of the matched patterns. Each token can match a set of patterns $\{P_{1..m}\}$, the size of this set is m. Normally not all patterns are valid for the corresponding token t_i, if ($p_j \in t_i$) then we can apply ($t_i - p_j$) = {''||$pref_j$, ''||$suff_j$}. Here the minus operator returns only the letters which form part of the token but are not included in the pattern, i.e. it returns a set of prefixes, suffixes, or nulls (''). Therefore, this process calculates all the possible prefixes and suffixes that exist in $t_i - p_j$. For the previous example the set of prefixes and suffixes are shown as follows:

(t_i - p_1)= (وعدوانها - وعد) = {‘‘, وانها *wAnhA*} where $pref_1$ = ‘‘, and $suff_1$ = وانها
(t_i - p_2)= (وعدوانها - وع) = {‘‘, دوانها *dwAnhA*} where $pref_2$ = ‘‘, and $suff_2$ = دوانها
(t_i - p_3)= (وعدوانها - وعدو) = {و *w*, انها *AnhA*} where $pref_3$ = و, and $suff_3$ = انها
(t_i - p_4)= (وعدوانها - عدوان) = {و *w*, ها *hA*} where $pref_4$ = و, and $suff_4$ = ها
(t_i - p_5)= (وعدوانها - وعدوا) = {و *w*, نها *nhA*} where $pref_5$ = و, and $suff_5$ = نها
(t_i - p_6)= (وعدوانها - عد) = {و *w*, وانها *wAnhA*} where $pref_6$ = و, and $suff_6$ = وانها

To reduce the set of matched patterns we have defined suffList that contains more than 200 suffixes normally used in the Arabic language. If $suff_j \notin$ suffList then the corresponding p_j is removed from the matched set. Therefore, for the previous example, the matched patterns p_1, p_2, and p_6 are removed from the set because {وانها *wAnhA*, دوانها *dwAnhA*, وانها *wAnhA*} are not suffix terms. If not all the calculated suffixes belong to suffList then the corresponding t_i is written as NoMatch (t_i) in TokensNotTagged file. The new matched patterns set will be {P’} and m’ = 3.

(g) EditDistance (t_i, {P’}): This process calculates the edit distance $d_j \in D$ between t_i and p_j, where D is the distances set. The edit distance is defined as (t_i.length()-p_j.length()). We use this distance to reduce the set of the matched patterns in order to make it easy to find the appropriate match. For the previous example, {$P'_{1.0.3}$} = {p_1 = عدو *wçd*, p_2 = عدوان *çdwAn*, p_3 = عدوا *çdwA*} and the set of the edit distances is D = {d_1 = 5, d_2 = 3, d_3 = 4) respectively. This means that p_2 could be the correct pattern of t_i because it has the minimum edit distance with t_i, while the other two patterns p_1 and p_3 are removed from the set.

(h) P–S Verification: Although in the previous processes (f and g) we carry out two types of verification to find the correct pattern, we have detected that we need more verification related to the compatibility between the prefixes and the suffixes with the entities. For example, the prefix (ال*Al*) is compatible with the noun category while it is not compatible with the verb category, and so on. If the prefixes or the suffixes are not compatible with the category type then the corresponding t_i is written as NoMatch (t_i) in the TokensNotTagged file. In Sect. 4.3, the prefix and suffix v e r i fication is explained in more details.

(i) Stemming (p_j): The main aim of this process is to reduce p_j to its root. The root extraction process is based on Khoja’s Stemmer (Khoja and Garside 1999). We adapted this stemmer according to our system needs. This process is applied to the selected patterns. The importance of root extraction for Arabic text mining is shown through the following table which shows examples (not all) of the multiple orthographic forms to write the Arabic word (زلزال*zlzAl*) ‘*earthquake’* according to their grammatical states. Each one of these forms shows the corresponding result in Google search.

بزلزال *bzlzAl*	191000	وزلزال *wzlzAl*	35200	الزلزاليه *AlzlzAlyh*	10900
لزلزال *lzlzAl*	202000	كالزلازل *kAlzlAzl*	45700	لزلازل *lzlAzl*	20000
والزلازل *wAlzlAzl*	584000	الزلزالية *AlzlzAlyħ*	50000	والزلزال *wAlzlzAl*	23900
بالزلزال *bAlzlzAl*	629000	الزلزالي *AlzlzAly*	57900	بزلازل *bzlAzl*	25500
الزلازل *AlzlAzl*	968000	بالزلازل *bAlzlAzl*	116000	وزلازل *wzlAzl*	27300
للزلازل *llzlAzl*	1000000	الزلزال *AlzlzAl*	133000	للزلزال *llzlzAl*	33200

When the word (زلزالzlzAl) *'earthquake'* is used in the plural, the orthographical form is changed to (زلازلzlAzl) *'earthquakes'*. This is due to what we call a broken plural where the singular form of the word is different from the plural form of the same word (Goweder et al. 2004). In addition, the different grammatical states such as: Prepositional, Nominative, Irregular, Non-standard, Accusative, etc. can add one or more letters to the original word. The root extraction process can help in the ED task because all the previous words have only one Arabic quadrilateral root (زلزلzlzl). Therefore, defining Arabic roots which are considered as interesting events for some applications will help to detect all the possible words referencing those events.

(j) XMLConveter (t_i, d_j, p_j, a_j, r): This process is used to convert the output information in XML file format. This information includes the original token, the distances between the token and the selected patterns, the attributes, and the root.

If there is no root for t_i then the symbol '-' is reported. Figure 3 shows an example of the XML output file. In this example the input text is T = (لمنظمة الامم المتحدة *lmnĎmħ AlAmm AlmtHdħ*) '*for the United Nations Organization*'. In this example, the matched pattern is منظمة امم متحدة *mnĎmħ Amm mtHdħ* with the attribute <a>Org. Nevertheless, the pattern امم متحدة *Amm mtHdħ* with the attribute <a>Org is also matched. What does this mean? It means that there is T' = (الامم المتحدة *AlAmm AlmtHdħ*) '*United Nations*' included in T that matches that pattern in ΣPMG. So our system returns all the patterns that would match all the input tokens in T or part of it. In addition, our system tags every word in T, so the token <t>لمنظمة </t> *lmnĎmħ* that consists of the Arabic prefix (ل *l*) 'for' and the word (منظمة *mnĎmħ*) 'organization'. This word matches منظم *mnĎm* with <a>مُنَظم |N|regulator;governor. These attributes are the morpho-

Fig. 3 An example of the classification of tokens

```
<tx>
            <t>لمنظمة</t>
            <p>منظم</p>
            <d>2</d>
            <a>مُنَظم|N|regulator;governor</a>
            <r>نظم</r>
</tx>
<tx>
            <t>الامم</t>
            <p>امم</p>
            <d>2</d>
            <a>أمَّم|PV|nationalize</a>
            <r>أمم</r>
</tx>
<tx>
            <t>المتحدة</t>
            <p>متحد</p>
            <d>3</d>
            <a>مُتَّحِد|N|United</a>
            <r>حدد</r>
</tx>
<tx>
            <t>الامم المتحدة</t>
            <p>امم متحدة</p>
            <d>4</d>
            <a>Org</a>
            <r>-</r>
</tx>
<tx>
            <t>لمنظمة الامم المتحدة</t>
            <p>منظمة امم متحدة</p>
            <d>5</d>
            <a>Org</a>
            <r>-</r>
</tx>
```

logical categories that depend on the BAMA (Buckwalter 2004). The Arabic word (مُنَظم*munaĎ ~ im*) shows the truly complete form of the matched pattern, and N denotes the token as a Noun. The rest of the attributes are the two possible glosses of the token in English. The <d>2</d> represents the edit distance between the token and the matched pattern. The <r>نظم</r> shows the trilateral root of the word. And so on for the other tags.

4.3 Prefix–suffix verification

In this section, we will explain our approach to managing the Arabic prefixes and suffixes to solve the compatibility problem of using them with each of our categories, and to minimize the impact of these affixes on the performance of our system. As we said, the Arabic prefixes and suffixes are particles added to the stems to obtain new grammatical states. To calculate the usage frequency of each of the Arabic prefixes and suffixes in a modern text, we have used an Arabic collection of documents with more than 12 million words. With regard to the prefixes, the results showed that almost 57.3% of the words in the collection did not take prefixes, while 12.9, 27.3, 2.4, and 0.03% of them did take prefixes of one, two, three, four letters respectively. With regard to the suffixes, the results showed that almost 70.7% of the words did not take suffixes, while 21.3, 7.3, 0.5, and 0.01% of them contained suffixes of one, two, three, four letters respectively. Table 2 shows some of the Arabic prefixes and suffixes with the highest frequency of use in the collection.

In the following, we will explain our approach to managing the compatibility of the Arabic prefixes and suffixes with each of our categories.

- Person Category {Per}: It contains more than 9,000 patterns of the Arabic first names and surnames. All these patterns have been defined exactly as they are spoken. Arabic first names and surnames do not normally take suffixes, so we do not need to verify the suffixes in this category. As regards the prefixes, in this category we consider that a prefix $pref_j$ can be formed from two particles

Table 2 Frequency of use of Arabic prefixes and suffixes

Prefixes	Frequency of use in the documents collection (%)	Suffixes	Frequency of use in the documents collection (%)
	57.3		70.7
الـ *Al*	24.82	ة *ħ*	15.75
و *w*	5.36	ا *A*	2.89
يـ *y*	2.49	ات *At*	2.59
بـ *b*	1.79	ه *h*	1.86
والـ *wAl*	1.57	ها *hA*	1.17
لـ *l*	1.50	ين *yn*	1.09
للـ *ll*	1.27	ون *wn*	0.55
تـ *t*	1.13	ته *th*	0.54
بالـ *bAl*	0.73	ن *n*	0.39
ويـ *wy*	0.44	هم *hm*	0.37

($pref_{j,0}$ + $pref_{j,1}$), $pref_{j,0}$ can be null or one of the following letters {و *w*, ب *b*, ل *l*, ف *f*, ك *k*}, while $pref_{j1}$ can be the definite article (ال *Al*) '*the*' which is the most common Arabic prefix. As an example, the prefix (وال *wAl*) '*and the*' consists of the conjunction article (و *w*) and (ال *Al*). According to our approach the first particle $pref_{j,0}$ can be used only with the first name, while the second particle $pref_{j,1}$ can be used with the surnames. Surnames are usually used with the particle $pref_{j,1}$ = (ال *Al*) like in *Aljumaily.* For this reason, when we define the Arabic surnames in the system dictionary we consider that $pref_{j,1}$ is a part of them. The verification of this category is carried out according to the following rules:

$\forall (t_i, p_j) \in \{Per\} \Rightarrow (t_i - p_j) \in (pref_{j,0} + pref_{j,1})$
$\Rightarrow pref_{j,0} \in \{$‘‘, ب *b*, ف *f*, ك *k*, ل *l*, و *w*$\}$ ^ $pref_{j,1} = \{$‘‘, ال$\}$

- Location Category {Loc}: It contains more than 2,000 patterns such as (continent, country, city, etc.) these patterns have been defined in the system dictionary without any prefixes. Normally the location names in Arabic do not take suffixes, thus the verification of the prefixes is performed according to the following rules:

$\forall (t_i, p_j) \in \{Loc\} \Rightarrow (t_i - p_j) \in pref_j$
$\Rightarrow pref_j \in \{$‘‘, وال *wAl*, بال *bAl*, فال *fAl*, لل *ll*, ال *Al*, و *w*, ب *b*, ل *l*, ول *wl*, وب *wb*, ف *f*, ك *k*$\}$

- Organization Category {Org}: It contains more that 1,000 patterns such as company, newspapers, web site, TV channel, etc., all these patterns have been defined in the system dictionary without any prefixes. Arabic organization names do not normally take suffixes. The following rule is used to verify this category:

$\forall (t_i, p_j) \in \{Org\} \Rightarrow (t_i - p_j) \in pref_j$
$\Rightarrow pref_j \in \{$‘‘, وال *wAl*, بال *bAl*, فال *fAl*, لل *ll*, ال *Al*, و *w*, ب *b*, ل *l*, ول *wl*, وب *wb*, ف *f*, ك *k*$\}$

- Noun Category {Nou}: It contains almost 50,000 nouns that have been defined without prefixes and suffixes. The verification of the prefixes and suffixes is performed according to the following rule:

$\forall (t_i, p_j) \in \{Nou\} \Rightarrow (t_i - p_j) \in \{pref_j, suff_j\}$
$\Rightarrow pref_j \in PN$ ^ $suff_j \in SN$ // where
PN = {‘‘, وال *wAl*, بال *bAl*, فال *fAl*, لل *ll*, ال *Al*, و *w*, ب *b*, ل *l*, ول *wl*, وب *wb*, ف *f*, ك *k*, ات *At*, ولل *wll*, وبال *wbAl*, ولل *wll*, كال *kAl*, وكال *wkAl*}
SN = {‘‘, ة *ħ*, ات *tA*, ا *A*, ين *yn*, ه *h*, ها *hA*, ن *n*, ي *y*, ون *wn*, ته *th*, هم *hm*, تها *thA*, ية *yħ*, اته Ath, نا nA, تين tyn, ان An, اتها AthA, مة mħ, تهم thm, تي ty, ى *ý*, و *w*, اتهم *Athm*, هما *hmA*, تنا *tnA*, كم *km*, يه *yh*, وا *wA*, تان *tAn*, ني *ny*, تهما *thmA*, هن *hn*, يها *yhA*, اتنا *AtnA*, وها *whA*, تك *tk*, تا *tA*, اي *Ay*, اتي *Aty*, اتهما *AthmA*, اه *Ah*, وه *wh*, كن *kn*, اتك *Atk*, تكم *tkm*, نه *nh*, اها *AhA*, نهم *nhm*, ناه *nAh*, وهم *whm*, نهن *nhn*, ما *mA*, اتهن *Athn*, انا *AnA*, ونا *wnA*, انه *Anh*, ونه *wnh*, تهن *thn*, نها *nhA*, تم *tm*, اكم *Akm*}

- Verb Category {Vrb}: It contains almost 32,000 verbs that have been defined without prefixes and suffixes in the system dictionary. The verification of the prefixes and suffixes in this category is carried out according to the following rule:

$\forall (t_i, p_j) \in \{Vrb\} \Rightarrow (t_i - p_j) \in \{pref_j, suff_j\}$
$\Rightarrow pref_j \in PV \wedge suff_j \in SV$ // where
PV = {“, و *w*, ي *y*, ت *t*, وي *wy*, ف *f*, سي *sy*, ن *n*, وت *wt*, ست *st*, سن *sn*, وسي *wsy*, لت *lt*, وب *wb*, في *fy*, ون *wn*, وا *wA*, ول *wl*, وست *wst*, م *m*, فت *ft*, لن *ln*, فسي *fsy*, فب *fb*, وسن *wsn*, فا *fA*, وسا *wsA*, فلي *fly*, فست *fst*, ولت *wlt*, ومت *wmt*}
SV = {“, ت *t*, ه *h*, ا *A*, ها *hA*, وا *wA*, ته *th*, ون *wn*, هم *hm*, نا *nA*, هما *hmA*, و *w*, ان *An*, وها *whA*, وه *wh*, تا *tA*, ونها *wnhA*, ني *ny*, اه *Ah*, تهما *thmA*, ناها *nAhA*, تني *tny*, تم *tm*, ناه *nAh*, تهن *thn*, نه *nh*, ونه *wnh*, وننني *wnny*, تاه *tAh*, تيهما *tyhmA*, ناكم *nAkm*, ناهم *nAhm*, نني *nny*, وهم *whm*, هن *hn*, وكم *wkm*, ونك *wnk*, ونكم *wnkm*, وننا *wnnA*, نكم *nkm*}

5 System evaluation

The Arabic language has become a popular area of research in IR, but it presents serious challenges due to its richness, complex morphology, and syntactic flexibility (Attia 2008). Although in the last decade a lot of work been carried out to make the task of NLP of Arabic easy, the systems to analyze Arabic automatically are not as easily available as they are for other languages such as English (Sawalha and Atwell 2008). In addition, available digital resources such as, for example, the corpora for Arabic NER are still limited although efforts are being made to remedy this (Farghaly and Shaalan 2009). One of the pioneering efforts is provided by the Linguistic Data Consortium (LDC) (2011), where more than one Arabic corpus annotated with regard to information extraction is provided in XML format.[2] Nevertheless, we have preferred to use ANERcorp (Natural Language Engineering Lab, http://users.dsic.upv.es/grupos/nle/?file=kop4.php) in our experiments because the Named entities of this corpus have been annotated on the text with a simple schema like (*token*$_1$*.tag*$_1$ *token*$_2$*.tag*$_2$ *token*$_3$*.tag*$_3$*...etc.*). This simple schema allows us to modify the corpus manually to include the two morphological categories considered in our work in addition to the Named entities (Person, Location, and Organization) provided in the original corpus. ANERcorp contains almost 150,000 terms which have been prepared from a collection of 316 articles that were manually obtained from various sources on the web.

In order to evaluate our approach, we applied widely used measurements such as Precision, Recall, Fmeasure, and Accuracy (Yang 1999). *Precision* is the ratio of the retrieved tokens which are relevant in the corpus, i.e., it evaluates the exactness of the system. A token is considered relevant when our tool labels it correctly with respect to the manually labeled corpus. *Recall* is the ratio of the retrieved relevant

[2] ACE 2004, 2005 Multilingual Training Corpus http://www.ldc.upenn.edu/Catalog/CatalogEntry.jsp?catalogId=LDC2005T09, http://www.ldc.upenn.edu/Catalog/CatalogEntry.jsp?catalogId=LDC2006T06.

tokens. It measures the ability of the system to retrieve a complete set of the relevant tokens from a corpus, i.e., it evaluates the system coverage. *Fmeasure* evaluates the effectiveness of the system. Accuracy shows the ability of the system to retrieve relevant tokens and discard irrelevant ones. The terms TruePositives (TP counts the tokens correctly assigned to this category), FalsePositives (FP counts the tokens incorrectly tagged to this category), FalseNegatives (FN counts the tokens incorrectly rejected from this category), and TrueNegatives (TN counts the tokens correctly rejected from this category) were gathered to calculate the previous measures, according to the following formulas:

$$Precision = \frac{TP}{TP + FP} \qquad Recall = \frac{TP}{TP + FN}$$

$$Fmeasure = \frac{2 * (\mathrm{Pr}\,cision * \mathrm{Re}call)}{(\mathrm{Pr}\,cision + \mathrm{Re}call)} \qquad Accuracy = \frac{TP + TN}{TP + FP + FN + TN}$$

To highlight the importance, and to clarify the effect of the Arabic prefixes and suffixes in the recognition results, we carried out two experiments to calculate the previous measures. The first experiment was carried out without Prefix–Suffix Verification (Sect. 4.3), while in the second one the verification process was used for all categories. Tables 3 and 4 show the results of the two experiments.

From these tables, we can distinguish the role of the Arabic prefixes and suffixes in each category. In general, the verification process improved the recognition results for all the categories although this improvement was not symmetrical. The biggest improvement was produced in the Verb category where the result of the precision has been improved from 51.19 to 85.79. This is because in the Arabic language adding some prefixes to the verbs produces other grammatical states. For example, the past verb (كتب*ktb*) 'he has *written*', changes to (أكتب*Âktb*) '*I write*', (يكتب*yktb*) '*he writes*', (تكتب*tktb*) '*she writes*', (الكتب*Alktb*) '*the books*' if it is associated with the prefix (أ*Â*, ي*y*, ت*t*, ال*Al*) respectively. We discuss the reason for this improvement by looking at the FP results of the Verb category in Tables 3 and 4, where the number of patterns was reduced from 17,150 to 2,762. It means that using the system without verification process returns many patterns as verbs, although these patterns are not labeled in the corpus. However, when the verification process was used, the number of these patterns was significantly reduced, so the precision result was increased.

Now, looking at Table 4, differences in precision and recall measures can be observed between categories. While Location precision is more than 81%, for the Organization category, precision reaches 65%. In our opinion, there are several reasons to explain these differences. The first one relates to the normalization process that our system carries out to remove the orthographical confusion problem (see Sect. 2). We think that this problem is difficult to resolve because it is part of the orthographic habits of the people. The second reason relates to the wide variations in usage and meaning of some of the Arabic words. In order to not overburden the reader, we will detail an example of it. The precision result of the Person category is (77.63%) which means that the system correctly tagged (5,425) tokens as persons, but at the same time, there are (1,563) tokens that are incorrectly tagged. This is because there are several Arabic words that have several meanings and uses, for

Table 3 The results of the first experiment (without Prefixes-Suffixes Verification)

	TP	FP	FN	TN	Precision	Recall	Fmeasure	Accuracy
Person	5,442	2,298	1,807	132,791	70.31	75.07	72.61	97.12
Location	2,999	950	1,784	136,582	75.94	62.70	68.69	98.08
Organization	1,124	737	1,084	138,670	60.40	50.91	55.25	98.71
Noun	39,593	14,305	8,530	86,633	73.46	82.27	77.62	84.68
Verb	17,987	17,150	1,416	105,394	51.19	92.70	65.96	86.92

Table 4 The results of the second experiment (with Prefixes–Suffixes Verification)

	TP	FP	FN	TN	Precision	Recall	Fmeasure	Accuracy
Person	5,425	1,563	1,813	133,543	77.63	74.95	76.27	97.63
Location	2,999	681	1,784	136,851	81.49	62.70	70.87	98.27
Organization	1,124	591	1,084	138,816	65.54	50.91	57.30	98.82
Noun	39,468	11,531	8,547	89,532	77.39	82.20	79.72	86.53
Verb	16,679	2,762	1,890	121,090	85.79	89.82	87.76	96.73

example words such as (سلام *slAm*) '*Salaam'*, (حياة *HyAħ*) '*Hayat'*, (تحسين *tHsyn*) '*Tahseen'*, etc., are used as people's first name, but these words are used also as nouns meaning '*peace*', '*live*' and '*improvement*' respectively. So, all these words have been returned as relevant to the Person category, although they were not labeled in the corpus. The third reason relates to the Latin words, where most of these words are written in Arabic in different ways. For example, the name of the Prime Minister of Spain '*Zapatero*' could be written in Arabic in the following ways: (ثاباتيرو *θAbAtyrw*, زاباتيرو *zAbAtyrw*, ثباتيرو *θbAtyrw*). If we search for these three words in Google, it returns the following number of documents (70,300, 31,200, 19,900) respectively. Our work does not currently provide the solution to this problem, so most of the tokens that were incorrectly rejected of this category were Latin words. The fourth reason can be observed in the case of the Organization category, where our corpus is about Arabic news, and normally the Arabic newspapers and web sites use names *such as; (*جريدة الانباء *jrydħ AlAnbA') '*Alanbaa Newspaper'*, (*الجزيرة نت *Aljzyrħ nt) '*Aljazeera *Net*', (جريدة المستقبل *jrydħ Almstqbl*) '*Almustaqbal Newspaper*', etc. Whenever such names are found in the corpus, they are correctly tagged as organization category, but often in the text of news only the most significant parts of these names are written. Therefore, this leads to more confusion because the system will tag these parts as nouns. The system does that because these parts of the names have other meanings such as; Alanbaa '*the news*', Aljazeera '*the island*', Almustaqbal '*the future*', etc.

The Recall/coverage results show the ability of the system to retrieve the complete set of the relevant tokens. In general, looking at Table 4, the Organization category has the lowest coverage because the system retrieved only (50.91%) of the relevant tokens in the corpus. The rest of the tokens have been lost because they were not defined in the system dictionary. The results for the rest of categories are almost the same.

6 Conclusion

In the last decade, the Arabic Language has become a popular area of research in IR in general and in text mining in particular. Unfortunately, working with Arabic adds more difficulties than the languages that derive from Latin, because it implies the solving of different types of problems such as the short vowels, *al-hamzah*, prefixes, suffixes, etc. In this work, we have tried to minimize the impact of the Arabic affixes on the performance of the system. A verification process has been designed to do that.

As we had expected, the verification process has improved the recognition results although this improvement was not symmetrical. The improvements in the results of Precision in all the categories Person, Location, Organization, Noun, and Verb are 7.32, 5.55, 5.14, 3.93, and 34.6, respectively. We do not think that these results are bad but they could be improved on. This depends on improving the following aspects in our system. (1) Improve the system dictionary by including new patterns in each category. (2) Another problem that must be solved in future works is how Latin words are written in Arabic because including all the possible ways of writing these words in the dictionary would be an impossible solution, so we need to improve our algorithms to detect all the possible ways used to write Latin words in Arabic. (3) Incorporate expert feedback to our system by using intelligent learning techniques such as active learning (Freund et al. 1997) or learning by demonstration techniques (Lieberman 2001) and (4) apply and adapt context analysis techniques already available to solve ambiguity issues for other languages, for example, this solution could help to clarify if (باريس*bArys*) 'Paris' is used as a Location or a Person in a sentence.

Acknowledgments This work has been partially supported by the Spanish Center for Industry Technological Development (CDTI, Ministry of Industry, Tourism and Trade), through the BUSCAMEDIA Project (CEN-20091026), and also by the Spanish research projects: MA2VICMR: Improving the access, analysis and visibility of the multilingual and multimedia information in web for the Region of Madrid (S2009/TIC-1542), and MULTIMEDICA: Multilingual Information Extraction in Health domain and application to scientific and informative documents (TIN2010-20644-C03-01). The authors would like also to thank the IPSC of the European Commission's Joint Research Centre for allowing us to include the EMM search engine in our system.

References

Abuleil, S. (2004). *Extracting names from Arabic text for question-answering systems. RIAO'04, Proceedings of the 7th international conference on Coupling approaches, coupling media, and COUPLING languages for information retrieval*, April 26–28, 2004 (pp. 638–647). France: University of Avignon (Vaucluse).

Afify, M., Sarikaya, R., Kuo, H.-K. J., Besacier, L., & Gao, Y. (2006). On the use of morphological analysis for dialectal Arabic speech recognition. Interspeech-2006, Pittsburg PA, September 2006.

Alqatta Alsaqly, I. (1999). *Building nouns, verbs, and Gerunds, reviewed by Dr. Ahmed Mohamed Abdel-Dayem.* Egypt: Dar Al Kutub.

Al-Sughaiyer, I., & Al-Kharashi, I. (2004). Arabic morphological analysis techniques: A comprehensive survey. *Journal of the American Society for Information Science and Technology, 55*(3), 189–213.

Al-Zoghby, A., Eldin, A. S., Ismail, N. A., & Hamza, T. (2007). Mining Arabic text using soft-matching association rules. In *International conference on Computer engineering & systems, 2007. ICCES '07* (pp. 421–426). November 2007.

Attia, M. (2008). *Handling Arabic morphological and syntactic ambiguities within the LFG framework with a view to machine translation*. PhD Dissertation, University of Manchester.

Benajiba, Y. (2009). *Arabic Named Entity Recognition*. PhD dissertation, Universidad Polite´cnica de Valencia.

Benajiba, Y., Diab, M., & Rosso, P. (October, 2008). Arabic Named Entity Recognition using Optimized Feature Sets. In *Proceedings of international conference on Empirical methods in natural language processing, EMNLP-2008* (pp. 284–293). Honolulu: Waikiki.

Benajiba, Y., Diab, M. T., & Rosso, P. (2009). Arabic Named Entity Recognition: A feature-driven study. *IEEE Transactions on Audio, Speech & Language Processing, 17*(5), 926–934.

Benajiba, Y., Rosso, P., & Benedı´, J. M. (2007). *ANERsys: An Arabic Named Entity Recognition system based on maximum entropy. Computational linguistics and intelligent text processing, 8th international conference*, February 18–24, 2007 (pp. 143–153). Mexico City: CICLing. Benajiba, Y., Zitouni, I., Diab, M., & Rosso, P. (2010). Arabic Named Entity Recognition: Using features extracted from noisy data. In *Proceedings of ACL 2010*, Uppsala, Sweden, July 2010.

Best, C., Steinberger, R., & Halkia, S. (2007). Web mining and intelligence. IPSC Institute for the Protection and the Security of the Citizen. Council of Europe. http://globesec.jrc.ec.europa.eu/publications/brochures/brochures/LB7606422ENC.pdf.

Buckwalter, T. (2004). *Buckwalter Arabic Morphological Analyzer Version 2.0*. LDC. http://www.ldc.upenn.edu/Catalog/CatalogEntry.jsp?catalogId=LDC2004L02.

DBpedia. (2009). http://dbpedia.org/About.

Diab, M. (2009). Second Generation Tools (AMIRA 2.0): Fast and robust tokenization, POS tagging, and base phrase chunking. In *MEDAR 2nd international conference on Arabic language resources and Tools*. Egypt: Cairo.

El potencial de la Red en a´rabe. Accessed April 27, 2010, from http://www.elmundo.es.

EMM search engine of the JRC. (2009). http://langtech.jrc.it/.

Farghaly, A., & Shaalan, K. (2009). Arabic natural language processing: Challenges and solutions. *ACM Transactions on Asian Language Information Processing, 8*(4), Article 14.

Freund, Y., Seung, H., Shamir, E., & Tishby, N. (1997). Selective sampling using the Query by Committee algorithm. *Machine Learning, 28*, 133–168.

GATE. (2009). A general architecture for text engineering. http://gate.ac.uk/.

Goweder, A., Poesio, M., & Roeck A. (2004). Broken plural detection for Arabic Information retrieval. In *Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval*, Sheffield, UK.

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

Habash, N., Rambow, O., & Roth, R. (2009). MADA + TOKAN: A Toolkit for Arabic tokenization, diacritization, morphological disambiguation, POS tagging, stemming and lemmatization. In *Proceedings of the 2nd international conference on Arabic language resources and tools (MEDAR)*, Cairo, Egypt.

Habash, N., Soudi, A., & Buckwalter, T. (2007). On Arabic transliteration. In A. van den Bosch & A. Soudi (Eds.), *Arabic computational morphology: Knowledge-based and empirical methods*. Berlin: Springer.

Halpern, J. (2007). The challenges and pitfalls of Arabic Romanization and Arabization. In *Second Workshop on Computational approaches to Arabic Script-based Languages (CAASL2)*. Stanford: Stanford University.

Internet World Stats, Usage and Population Statistics. http://www.internetworldstats.com/.

Kaye, A. S. (1991). The Hamzat al-Wasl in Contemporary modern standard Arabic. *Journal of the American Oriental Society, 111*(3), 572–574. http://www.jstor.org/stable/604273.

Khoja, S., & Garside, R. (1999). *Stemming Arabic text. Computing department.* Lancaster U.K.: Lancaster University. Accessed September 22, 1999, from http://www.comp.lancs.ac.uk/computing/users/khoja/stemmer.ps.

Khoja, S., Garside, R., & Knowles, G. (2001). A tagset for the morphosyntactic tagging of Arabic. In *Paper given at the Corpus Linguistics 2001 conference*, Lancaster.

Lieberman, H. (Ed.). (2001). *Your wish is my command: Programming by example*. San Francisco, CA: Morgan Kaufmann.

Linguistic Data Consortium (LDC). (2011). http://www.ldc.upenn.edu/.

Maamouri, M., et al. (2009) *LDC Standard Arabic Morphological Analyzer (SAMA) Version 3.1*. http://www.ldc.upenn.edu/Catalog/CatalogEntry.jsp?catalogId=LDC2010L01.

Martin, A., & Van der Goot, E. (2009). Near real time information mining in multilingual news. In *Proceedings of the 18th international World Wide Web conference (WWW'2009)*, Madrid, 20–24 April 2009 (pp. 1153–1154). New York: ACM.

Natural Language Engineering Lab. http://users.dsic.upv.es/grupos/nle/?file=kop4.php.

Pouliquen, B., & Steinberger, R. (2007). C. Automatic detection of quotations in multilingual news. In *Proceedings of the international conference recent advances in Natural language processing (RANLP'2007)*, 27–29 September 2007(pp. 25–32). Borovets, Bulgaria.

Sawalha, M., & Atwell, E. (2008). Comparative evaluation of Arabic language morphological analysers and stemmers. In *Proceedings of COLING 2008 22nd international conference on Computational linguistics*.

Shaalan, K. F., & Raza, H. (2008). Arabic Named Entity Recognition from diverse text types. In *Advances in natural language processing, 6th international conference, GoTAL 2008*, Gothenburg, Sweden, August 25–27, 2008, Proceedings. Lecture Notes in Computer Science 5221 (pp. 440–451). Berlin: Springer, ISBN 978-3-540-85286-5.

Silberztein, M. (2002). *NOOJ: A cooperative object oriented architecture for NLP*. I n *5th INTEX Workshop, May 2002*, Marseille, France.

Steinberger, R., Pouliquen, B., & Ignat, C. (2008). Using language-independent rules to achieve high multilinguality in text mining. In F. Fogelman-Soulie´, P. Domenico, J. Piskorski, & R. Steinberger (Eds.), *Mining massive data sets for security* (pp. 217–240). Amsterdam: John Benjamins Publishers.

The Arab League Educational, Cultural and Scientific Organization (Alecso). (2007). http://www.alecso.org.tn/.

The World Wide Web Consortium (W3C). (2009). http://www.w3.org/.

Yang, Y. (1999). An evaluation of statistical approaches to text categorization. *Journal of Information Retrieval, 1*(1/2), 67–88.