I.J. Information Technology and Computer Science, **2016, 2, 1-13**
Published Online February 2016 in MECS (http://www.mecs-press.org/)
DOI: 10.5815/ijitcs.2016.02.01

A Semi-Automatic and Low Cost Approach to Build Scalable Lemma-based Lexical Resources for Arabic Verbs

Noureddine Doumi
Computer Science Dept., University of Saïda, Algeria
E-mail: noureddine.doumi@univ-saida.dz

Ahmed Lehireche
Computer Science Dept., University of SBA, Algeria
E-mail: elhir@univ-sba.dz

Denis Maurel
Université François Rabelais Tours, LI computer laboratory, France
E-mail: denis.maurel@univ-tours.fr

Ahmed Abdelali
Qatar Computing Research Institute, Qatar
E-mail: aabdelali@qf.org.qa

***Abstract*—**This work presents a method that enables Arabic NLP community to build scalable lexical resources. The proposed method is low cost and efficient in time in addition to its scalability and extendibility. The latter is reflected in the ability for the method to be incremental in both aspects, processing resources and generating lexicons. Using a corpus; firstly, tokens are drawn from the corpus and lemmatized. Secondly, finite state transducers (FSTs) are generated semi-automatically. Finally, FSTs are used to produce all possible inflected verb forms with their full morphological features. Among the algorithm's strength is its ability to generate transducers having 184 transitions, which is very cumbersome, if manually designed. The second strength is a new inflection scheme of Arabic verbs; this increases the efficiency of FST generation algorithm. The experimentation uses a representative corpus of Modern Standard Arabic. The number of semi-automatically generated transducers is 171. The resulting open lexical resources coverage is high. Our resources cover more than 70% Arabic verbs. The built resources contain 16,855 verb lemmas and 11,080,355 fully, partially and not vocalized verbal inflected forms. All these resources are being made public and currently used as an open package in the Unitex framework available under the LGPL license.

***Index Terms*—**Arabic NLP, Arabic linguistic resources, Arabic verbs, Finite state transducers, Unitex.

I. Introduction

The Arabic is a member of the Semitic language family consisting mainly of Arabic, Aramaic, Amharic and Hebrew. Semitic languages are characterized by i) a lexicon built mainly from triliteral and quadriliteral roots ii) a writing system from right to left and iii) an alphabet of Abjed nature where only consonants, not vowels, are represented among the basic graphemes [1]. The Arabic is the first Semitic language in number of speakers with more than 340 million speakers and the fourth World language in number of Internet users[1].

Arabic is divided into Classical Arabic (CA) and Modern Standard Arabic (MSA), the former being the language of sacred texts of Islam, the Koran and the Hadith and also the language of the cultural, literary and scientific heritage of the Arab-Muslim civilization. The MSA is the direct descendent of the former and is the official language of the Arab world today; so it is used in education, media and administrative correspondence. MSA expressed differences from CA on the lexical, morphological and syntactic levels [2, 3]. These differences were summarized by Attia [3] in the following points:

- The MSA lexicon is richer than CA lexicon, since it incorporates new words borrowed from other languages,
- In general, MSA conforms to the morphology and syntax rules of CA, but in MSA there is a greater tendency for simplification and modern writers use only a subset of the full range of structures, inflections and derivations available in CA,

[1] These statistics are from the Internet World Stats of 2013 and can be found at http://www.internetworldstats.com (last accessed Dec 2014)

- The classical word order of OVS is rarely found in MSA,
- In MSA, there is a tendency to avoid passive verb forms where the active readings are also possible,
- The relatively marginal SVO word order in CA is gaining more weight in MSA.

Used by over 22 countries worldwide, Arabic vernacular is classified into seven groups [4]: Egyptian Arabic (EGY), Levantine Arabic (LEV), Gulf Arabic (GLF), Maghreb Arabic (MAG), Iraqi Arabic (IRQ), Yemeni Arabic (YEM) and Maltese Arabic (MLT).

Processing Arabic like other languages requires large linguistic resources in order to carry different tasks. Whether the task is lightweight such as spellchecker or heavy like machine translation, these resources are crucial. Lexica are recognized as a fundamental pre-requisite for all natural language processing tasks [5]. Building one's own linguistic resources is an economic way for the NLP researcher to acquire a crucial component in these tasks. On the other hand, it is quite difficult and beyond the budget of the researchers to build wide-scale coverage resources for the target language; because one the most challenging factors is that we are nowhere close to constructing a complete and systematic dictionaries of the basic linguistic building blocks for majority of languages if not for all. Such precious resources could be put to immediate use in a large variety of applications such as automatic translation, information retrieval, syntactic parsing and many other areas [5]. For this reason the current paper suggests a method for building scalable resources for Arabic verbs. It should be noted that Arabic verbs are monolexical units; hence we focus only on monolexical unit dictionaries. In the next section, we survey related work concerning the Arabic lexical resources; then a theoretical background on finite-state technology and Arabic morphology is introduced in section three. In the fourth section, we outline the details and objectives of the proposed approach. Finally we present the results of the experimentation of our system and the produced transducers and then we evaluate them against the gold reference.

II. Related Work

Different techniques were used to automatically acquire lexica from corpora of different languages, e.g. for French, the key insight in [6] relies on the idea that the existence of a hypothetical lemma can be guessed if several different words found in the corpus are best interpreted as morphological variants of this lemma.

In [7] the authors extract automatically a bilingual Arabic-English dictionary from a parallel corpora, the process of the approach takes four stages : preprocessing, alignment, extracting and filtering. They apply different tools and techniques to achieve these four steps, the MADA [8] was used in the preprocessing step, the GIZA++ [9] for alignment and Pharaoh [10] for translation extraction and finally they used Ripper [11] a rule-based machine learning classifier to accomplish the filtering task.

The work of [12] presents an inference algorithm that organizes observed words (tokens) into structured inflectional paradigms (types). It also naturally predicts the spelling of unobserved forms that are missing from these paradigms, and discovers inflectional principles (grammar) that generalize to wholly unobserved words.

Durret in [13] presents a completely data-driven and language independent approach; the paper describes a supervised technique to predicting the set of all inflected forms of a lexical item. The system automatically acquires the orthographic transformation rules of morphological paradigms from labeled examples, and then learns the contexts in which those transformations apply using a discriminative sequence model.

In the rest of this section we will survey work related to Arabic lexica and try to emphasize the characteristics of each resource with respect to our work. The work cited below is not necessarily carried out using the finite-state machines.

2.1 BAMA/SAMA Lexicon

The Buckwalter Arabic Morphological Analyzer (BAMA) [14] is widely used in the Arabic NLP research community. It is designed for analysis and not generation and therefore its lexical resources take the stem-affixes format. The verbal resource contains 8,709 lemmas and 33,393 stems, each verb may have up to five stems matching the tenses of perfect active, imperfect active, perfect passive, imperfect passive and imperative [3, 15]. The entire resources cover 40,648 Arabic lemmas and over 82,000 stems, these items are structured in three tables A, B and C. An Arabic word is considered as a concatenation of prefix, stem and suffix. Sublexicon A contains all the combinations of proclitics and inflectional prefixes for verbs and nouns (561 items), sublexicon C contains all the combinations of inflectional suffixes and enclitics for verbs and nouns (989 items). Table B contains the lemmas and their corresponding stems.

The most recent version of BAMA was released under the name of SAMA (Standard Arabic Morphological Analyzer) [16]. Attia [3] and Neme [15] list some drawbacks of SAMA lexicon which we summarize as follows:

- 25% of the lexical items are obsolete,
- Lexical resources of SAMA are not representative of MSA,
- Although SAMA resources are open, it is complex to extend it with new entries,
- The stem lexicon entries corresponding to a lemma are numerous and need to be subcategorized.

2.2 Aracomlex Lexicon

For the construction of a lexicon for MSA, Attia [3] took advantage of large and rich resources that have not been exploited in similar tasks before. They used a corpus of 1,089,111,204 words, consisting of 925,461,707 words from the Arabic Gigaword corpus, fourth edition, in

addition to 163,649,497 words from news articles collected from the Al-Jazeera web site. Then the corpus is pre-annotated using MADA, a state-of-the-art tool for morphological processing. MADA combines SAMA and SVM classifiers to choose the best morphological analysis for a word in context, doing lemmatization, diacritisation, POS tagging and disambiguation at the same time with high accuracy.

In this work, Attia et al. have overtaken the disadvantage of SAMA by using MADA and a data-driven filtering approach to identify core MSA lexical entries rather than obsolete words.

The result lexicon reduced the SAMA entries from 40,648 lemmas to 29,627 lemmas with a rate of 72.89%. The original number of verb lemmas was 8,709 entries, the new number of verb lemmas may be estimated around 6,350 lemmas.

2.3 DIINAR

The DIINAR project was developed in Lyon2 University for terminological and translation purposes. The total number of lemma entries in the DIINAR.1 database equals 121,522. This includes 445 tool-words belonging to various grammatical categories (e.g.: prepositions, conjunctions, etc.) and the prototype of a proper name database of 1,384 entries. Both types of entries are associated with a particular word-formative grammar, and with their own subsets of morpho-syntactic specifiers [17].

The entries are fully vocalized and include 19,457 verb lemmas. A conventional programming framework and databases are used for generation and analysis with a lemma-based lexicon encoded according to this framework [15].

Even though they have the highest coverage percentage of all Arabic lexical resources, the DIINAR resources remain not open and they are out of reach for researchers with a cost of €11,000.

2.4 Almorgeana and Elexir Lexicon

Both of the projects Almorgeana [8] and ElexirFM [18] extend the BAMA with the generation ability. They are very close in spirit because both of them implement the functional Arabic morphology [4]. The building of Almorgeana didn't just involve the reversal of the Buckwalter analyzer engine, which only focuses on analysis, but also extending it and its databases to be used in a lexeme-and-feature level of representation for both analysis and generation.

The lexicon of ElexirFM project is derived from the open-source Buckwalter lexicon and is enhanced with information from the syntactic annotations of the Prague Arabic Dependency Treebank.

Functional Arabic Morphology is a formulation of the Arabic inflectional system seeking the working interface between morphology and syntax. ElixirFM is its high-level implementation that reuses and extends the Functional Morphology library for Haskell. Inflection and derivation are modeled in terms of paradigms, grammatical categories, lexemes and word classes. The computation of analysis or generation is conceptually distinguished from the general-purpose linguistic model. The lexicon of ElixirFM is designed with respect to abstraction, yet is no more complicated than printed dictionaries [18].

2.5 ALESCO Sarf System

The Sarf system of ALESCO [19] is a derivation and inflection system for Arabic and is based on root-and-pattern representation. This work has the advantage of being clearly built on a strong linguistic basis that is the standard morphology in Arabic [15]. As far as we know the Sarf lexicon has the best coverage in terms of roots, verbs and derivative nominals, compared to the work previously done both open or proprietary.

In this project the lexical materials are acquired from the reference books of Arabic lexicography; from CA dictionaries such as al-Muheet by al-Sahib bin 'Abbad (died 995), al-Sihah by Ismail ibn Hamad al-Jawhari (died 1009), Lisan al-'Arab by ibn Manthour (died 1311), al-Qamous al-Muheet by al Fairouzabadi (died 1414) and Taj al-Arous by Muhammad Murtada al-Zabidi (died 1791) and from the most kwon dictionaries of MSA such as Muheet al-Muheet (1869) by Butrus al-Bustani and al-Mu'jam al-Waseet (1960) by the Academy of the Arabic Language.

The number of 7,564 roots gathered in this project, represents almost the entire Arabic language roots. The number of derived verbs is 21,705 triliteral lemmas and 2,308 quadriliteral lemmas which represents over 24,000 verb lemmas. From these roots the Sarf system can generate all derivative nouns, gerunds and adjectives.

Despite its good coverage, this project doesn't include the primitive nominals and gerunds.

2.6 NooJ Arabic Lexicon

The NooJ NLP platform [20] is a natural language processor; it regroups the linguistic resources for several languages. Its Arabic lexicon is lemma-based and counts 10,500 fully vocalized verbs [21]. The project does not use root-and-pattern representation; the author has suggested a new classification of Arabic verbs. The inflection and derivation of these verbs are accomplished by finite state transducers; each transducer represents a derivational or inflectional paradigm. The classification consists of 125 derivational paradigms and 130 inflectional paradigms. Neme [15] indicates that there are no figures on testing and evaluating the available system.

III. THEORETICAL BACKGROUND

In the remainder of this paper we will focus on finite state technology and we will highlight the use of this technology in NLP in general and its use in building and storing the Arabic electronic dictionaries. So it is necessary to bring the formal definition of related terminology. FSM (Finite-State Machine) is a generic term to design all types of automata.

Definition 1

An finite-state automaton (FSA) over $\mathcal{L}$ is a five-tuple $\mathcal{A}=(\mathcal{Z},\mathcal{L},q_0,\mathcal{F},\delta)$ where :

- $\mathcal{Z}$ is a non-empty finite set of states
- $\mathcal{L}$ is non-empty set of letters and diacritics (the alphabet)
- q_0 is an element of $\mathcal{Z}$ (the initial state)
- $\mathcal{F}$ is a non-empty subset of $\mathcal{Z}$ (the final states)
- δ is a relation defined from $\mathcal{Z}$ x $\mathcal{L}$ to $\mathcal{Z}$ (the transitions).

We say that an element $\ell_1\ell_2\ldots\ell_n$ is recognized by the automaton $\mathcal{A}$ if and only if it corresponds to a sequence of n transitions labeled respectively by ℓ_1, ℓ_2,… and ℓ_n, beginning with the initial state q_0 and ending with a final state p_n

$\ell_1\ell_2\ldots\ell_n$ is recognized by $\mathcal{A}$ iff:

$$\exists\, p_0, p_1, \ldots, p_n \in \mathcal{Z}, \text{ such that :}$$
$$\forall\ i=0,1,\ \ldots,\ n-1,\ p_{i+1} \in \delta(p_i, \ell_{i+1})$$
$$p_0=q_0$$
$$p_n \in \mathcal{F}$$

We call $\mathcal{L}^*$ the set consisting of the empty ε string and all finite sequences of elements of $\mathcal{L}$. The set of elements in $\mathcal{L}^*$ recognized by $\mathcal{A}$ is called the *language* defined by $\mathcal{A}$. it is designated by L($\mathcal{A}$).

In the case of MSA, the alphabet set $\mathcal{L}$ consists of the Unicode range from \U0621 to \U0652. The basic Arabic range encodes the standard letters and diacritics, but does not encode contextual forms[2].

For instance, the Fig.1 represents FSA recognizing/storing the Arabic week days, the transitions are labeled only by consonants and long vowels. In order to simplify and to get non cumbersome figure we consider that words recognized by this automaton are unvoweled[3] (without short vowels or gemination mark).

Definition 2

A finite-state-p-subsequential transducer (FST) over L is a seven-tuple T = (Q, L, S, q_0, q_1, δ, λ)

- $\mathcal{Z}$ is a non-empty finite set of states
- $\mathcal{L}$ and $\mathcal{S}$ are non-empty finite sets of Arabic and Latin letters and diacritics (the input and the output alphabets, respectively)
- q_0 is an element of $\mathcal{Z}$ (the initial state)
- q_f is an element of $\mathcal{Z}$ (the final state)
- δ is a function defined from $\mathcal{Z}$ x $\mathcal{L}$ to $\mathcal{Z}$ (the transition function)
- λ is a function defined from $\mathcal{Z}$ x $\mathcal{L}$ to $\mathcal{S}^*$ (the transition output function)

The automaton ($\mathcal{Z}$, $\mathcal{L}$, q_0, q_f, δ) is called the underlying FSA of the transducer $\mathcal{T}$, for instance, the Fig. 2 shows an example of a transducer recognizing the Arabic week days and informing for each day whether it is a working day or day off. The FSA of Fig. 1 is acyclic and nondeterministic; this kind of automaton is used to recognize/store the lexicon of natural languages and sometimes termed lexicographic tree. The underlying automaton of the transducer in Fig. 2 is the result of determinization and minimization of this lexicographic tree.

Definition 3

An automaton $\mathcal{A}$ = ($\mathcal{Z}$,$\mathcal{L}$,q_0,$\mathcal{F}$,δ), without ε-transitions, is called deterministic if and only if all the transition labels from any state are distinct. This is equivalent to noting that δ is a function defined from $\mathcal{Z}$ x $\mathcal{L}$ to $\mathcal{Z}$.

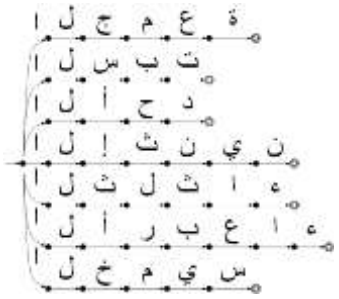

Fig.1. Lexicographic Tree or Acyclic Nondeterministic Automaton Recognizing/Storing the Arabic Week-Days

Theorem

Given a deterministic automaton $\mathcal{A}$, there exists one and only one minimal deterministic automaton $\mathcal{A}'$ which recognizes the same language [22].

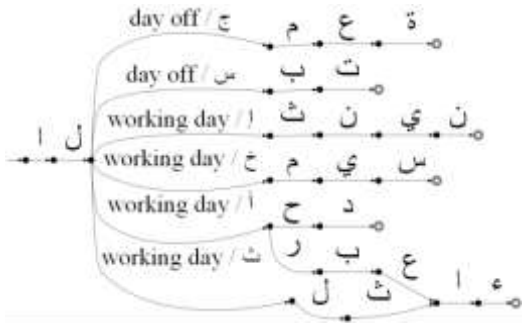

Fig.2. Transducer Recognizing the Arabic Week-Days and Informing if they are Working Day or Day off. The Underlying Automaton is the Determinization and Minimization of Automaton of Fig.1.

The minimization consists of eliminating the redundant states and transitions in automaton. It looks at first sight like a kind of reverse determinization [5]. For instance, in

[2] For instance the Unicode table lists over 1260 Arabic characters for the Arabic script used even in other languages such as for Persian, Urdu, Sindhi and Central Asian languages etc. In MSA the regarded alphabet isL = {ء, آ, أ, ؤ, إ, ئ, ا, ب, ة, ت, ث, ج, ح, خ, د, ذ, ر, ز, س, ش, ص, ـ, ف, ق, ك, ل, م, ن, ه, و, ى, ي, ً, ٌ, ٍ, َ, ُ, ِ, ّ, ْ}

[3] The length of fully voweled words is usually the double of unvoweled ones

Fig.2 the underlying automaton of the presented transducer is the result of determinization and minimization of non-deterministic acyclic automaton (lexicographic tree) of Fig.1. We can easily notice that the number of states is reduced at 60% of states of the lexicographic tree.

3.1 Finite State Machines in NLP

Finite state technologies have shown their capacity to model different phenomena of natural language and their use has been shown to be successful in various areas of computational linguistics: lexical analysis, morphology and phonology, local syntax, syntax, text-to-speech synthesis and speech recognition [23]. Theoretically the finite-state automata are interesting because they are highly constrained; and in practical computational linguistics for natural language, finite-state automata are fast, usually compact in size, bidirectional, combinable using all valid finite-state operations and consultable using language-independent lookup code [24].

3.2 Dictionaries as Finite State Machines

In finite state calculus, handling large automata of lexica with their inflections that can run into millions of paths is a matter of seconds [3]. The works of [25-34] show that minimal acyclic deterministic finite state automata represent finite languages and are therefore useful in applications such as storing words for spell checking, computer and biological virus searching, text indexing and XML tag lookup. In such applications, the automata can grow extremely large (with more 10^6 states) and are difficult to store without compression or minimization [35]. The experiments show the usefulness of the p-subsequential transducers and their minimization in the representation of large scale dictionaries [23]. A transducer is in general less voluminous than a multi-terminal automaton (sometimes termed Moore automaton) [5]. Like deterministic automata, sequential transducers provide a very fast look-up depending only on the length of the input string and not on the size of the machine [23].

Maurel and Guenthner [5] have studied the temporal and spatial complexity of storing a dictionary in a finite-state machine. In this study they compares a dictionary of *D* characters and *W* words stored in text file and minimal deterministic acyclic automaton of *S* states and alphabet *A* constructed from this dictionary. the study has concluded that the memory space is O(*D*) for the dictionary and O(*S*) for the automaton and the time needed to read a given word of length *w* is O($W \times w$) for the dictionary and is O($A \times w$) for automaton. The construction time of the automaton is estimated by O($D \times \log(S)$)[4].

3.3 Arabic Inflection and Derivation

Derivational morphology is concerned with creating new words from other words, a process in which the core meaning of the word is modified. For instance, the Arabic كاتب/kAtib/[5]*writer* can be seen as derived from the verb كتب/katab/*to write* the same way the English noun"writer" can be seen as a derivation from the English verb "write" [4]. Derivational morphology usually involves a change in POS. The derived variants in Arabic typically come from a set of relatively well-defined lexical relations, e.g., location (اسم مكان/Ism makAn/), time (اسم زمان/Ism zamAn/), actor/doer/active participle (اسم فاعل/Ism fAçil/) and actee/object/passive participle (اسم مفعول/Ism mafçowl/) among many others. The derivation of one form from another typically involves a pattern switch. In the example above, the verb كتب /katab/ has the root-ك-ت-ب k-t-b and the pattern 1a2a3; to derive the active participle of the verb, we switch to the pattern 1A2i3 to produce the form كاتب/kAtib/*writer*.

Although compositional aspects of derivations do exist, the derived meaning is often idiosyncratic. For instance, the masculine noun مكتب/maktab/*office/bureau/agency* and the feminine noun مكتبة/maktaba.h/ *library/bookstore* are derived from the root ك-ت-ب k-t-b *writing-related* with the pattern+vocalism ma12a3, which indicates location. The exact type of the location is thus idiosyncratic, and it is not clear how the nominal gender difference can account for the semantic difference.

On the other hand, in inflectional morphology, the core meaning and POS of the word remain intact and the extensions are always predictable and limited to a set of possible features. Each feature has a finite set of associated values.

For instance, in the morphological analysis of the word وكتبه/wakutubuhu/ may have 2 cases:

|katab|,Verb, conjunction:wa, particle:ø, article: ø, person:3rd, gender:masc, number:sing, case:n/a, aspect:perfect,object:3MS, *and he wrote it*

|kitAb|, Noun, conjunction:wa, particle:ø, article: ø, person:n/a, gender:masc, number:plur, case:gen, aspect:n/a, possessive:3MS, *and his books* [genitive]

In the second case, the feature-value pairs number:plur and case:gen, indicate that particular analysis of the word وكتبه/wakutubihi/ is plural in number and genitive in case, respectively. Inflectional features are all obligatory and must have a specific (non-null) value for every word. Some features have POS restrictions. In Arabic, there are eight inflectional features. Aspect, mood, person and voice only apply to verbs, while case and state only apply to nouns/adjectives. Gender and number apply to both verbs and nouns/adjectives.

3.4 Transducers for Arabic Derivation and Inflection

There are a number of advantages of the finite state technology that makes it especially attractive in dealing with human language morphologies; among these are the ability to handle concatenative and non-concatenative

[4] This time complexity is estimated by the algorithm of Daciuk and Mihov [36].

[5] Here and in what follows we use HSB transliteration (a variant of the Buckwalter transliteration) following every Arabic word. The HSB transliteration is chosen instead of the former for the following reason: one of the common critiques of the Buckwalter transliteration is that it is not easy to read [4].

morphotactics, as well as high speed and efficiency. Just like other languages, different research work in Arabic NLP have shown that FSM can handle Arabic linguistic rules even though the Arabic is an agglutinative and non concatenative language [37].

As in Definition 2 a finite-state transducer is a finite automaton whose state transitions are labeled with both input and output symbols. Then it gives a result sequence of letters in output when recognizing the input one. Hence generally in Semitic languages and particularly in Arabic, the transducers are used in the derivation as well as in inflection. In the derivation the transducer produces verb or noun lemma when the root is recognized while in the inflection it produces all possible word forms [6] combined with their full morphological features when the lemma is recognized.

The Arabic verb lemma is derived from the triliteral and quadriliteral roots. As it is stated in [38] we can derive more than 24,000 Arabic verb lemmas from over 7,500 Arabic roots[7] by applying derivation patterns on the root consonants. For instance, to derive the triliteral primitive verb [8] lemma the following algorithmic rule, exposed in [19], is used:

$$\text{Verb}(i) = \text{C1} + \text{ـَ} + \text{C2} + \text{VPA2}(n) + \text{C3} + \text{VPA3}(i) + \text{ConPro1}(i), \text{with} 1 \leq i \leq 13 \text{ and } 1 \leq n \leq 6$$

To calculate the lemma, in this rule the *i* must equals 3 and *n* can take values from 1 to 6. The transducer of Fig.4 recognizes a triliteral root and outputs one of these 6 possibilities. The upper path of the transducer outputs the lemma of the derived verb which matches in Arabic the 3rd person masculine of the past tense form. The lower path generates the 3rd person masculine of the present tense form. The combination of the two word forms allows identifying the 6 derivation patterns. For example, the transducer of Fig.4 represents the pattern 1a2a3a-ya12u3u (يَفْعُلُ-فَعَلَ/façala-yaf.çulu/).

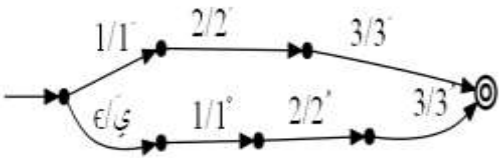

Fig.3. Derivation Transducer Producing Triliteral Primitive Verbal Lemma Derived from a Triliteral Root.

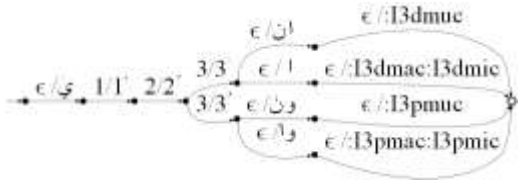

Fig.4. Inflection Transducer Producing Six Verbal Word forms of Triliteral Primitive Verb Lemma. Each Word Form Is Combined with its Inflectional Features.

IV. The Proposed Approach

The building of lexical resources has close relationship with morphology generation. And the Arabic morphology concerns four phenomena: derivation, inflection, cliticization and morphophonemic/orthographic adjustments [4]. Because the nature of our lexical resources (lemma-based) we applied only inflection and morphophonemic/orthographic adjustment rules in building our verbal word form lexicon.

Our proposed approach is not data-driven but knowledge-based and hand-engineered technique. The rules are written as FSMs (part of the rules are hard-coded in conventional programming language) and the generated lexica are stored as FSMs. Although the proposed approach is not corpus-based we use a representative corpus of MSA for two reasons : i) The statistics of Arabic shows that the number of Arabic verb roots is estimated for more than 7,000 but the used ones do not exceed 1,000 [39]. So in our algorithm, instead of remembering all the verbs and adding them to the dictionary, we suggest adding only those having place in current usage. ii) The existing Arabic traditional dictionaries make no distinction between entries from MSA and CA. Therefore, they tend to include obsolete words that have no place in current usage. The current computational resources have inherited this drawback; for example SAMA lexicon contains several thousands of entries that are hardly ever encountered by modern Arabic speakers [3]. For these reasons, in our work the words attested in the MSA data i.e. encountered in the corpus are included in the lexicon while the others are filtered out.

4.1 Built Resources

As mentioned above, the target resources are mono-lexical unit dictionaries for Arabic verbs. And to comply with Unitex corpus processor we choose the DELA (Dictionnaires Electroniques du LADL, LADL [9] electronic dictionaries) structure as a format [40, 41]. The DELA dictionaries list practically all observed elementary or simple forms together with the relevant information about their inflectional paradigms. The basic form of such dictionaries is always the same and contains, for the moment, the following types of information : <Full form, Lemma, Syntactic category, Morphological

[6] Here and in what fellows we use *word form* to mean the *inflected form* without clitics. We assume that the system using our dictionaries must have a processing unit to remove the proclitics and enclitics before applying our dictionaries.

[7] The number of Arabic triliteral roots is more than 5734 and the quadriliteral roots are estimated more than 1830 roots. The number of derived verbs from triliteral roots is 7889 primitive verbs and 13816 augmented verbs. The number of derived verbs from quadriliteral roots is 1460 primitive verbs and 848 augmented verbs [38].

[8] Primitive or basic as opposed to augmented verbs.

[9] Acronym of *Laboratoire d'Automatique Documentaire et Linguistique* cf. http://infolingu.univ-mlv.fr/LADL/Historique.html

codes> [5].

Examples:
Arabic: <يَكْتُبُونَ,كتب,V:I3pmc>
English: <dances,dance.V:P3> <dances,dance.N:p>
French: <danses,danser.V:P2s:S2s> <danses,danser.N:p>
In the given examples we used morphsyntactic features of our tag set [10]. This tag set is the result of a deep comparative study of several tag sets of previous Arabic NLP research projects. For further details we refer the reader to [1].

4.2 DELA of LADL Formalism

Unitex is a corpus processing system based on automata-oriented technology. The electronic dictionaries distributed with Unitex use the DELA syntax. This syntax describes the simple and compound lexical entries of a language with their grammatical, semantic and inflectional information. We distinguish two kinds of electronic dictionaries. The one that is used most often is the dictionary of inflected forms DELAF (DELA of inflected forms) or DELACF (DELA of compound inflected forms) in the case of compound forms. The second type is a dictionary of canonical forms called DELAS (simple forms DELA) or DELAC (compound forms DELA).

Unitex programs make no distinction between simple and compound form dictionaries. We will use the terms DELAF and DELAS to distinguish the inflected and non-inflected dictionaries, no matter they contain simple words, compound words or both.

V. METHODOLOGY

In this section, we explain the techniques we followed in the construction of our lexical resources.

5.1 Overview of the Methodology

The aim of the methodology is enabling the Arabic NLP community to build their own scalable lexical resources, by producing new entries in both lemma and word form dictionaries: DELAS and DELAF. Fig.5 summarizes this methodology and the subsequent paragraphs give further details. Algorithm 1, algorithm 2, algorithm 3 and algorithm 4 give the details about the automatic part of the methodology.

To add a new verb to the dictionary the user/annotator[11] is guided by a cursor in the text. So he browses through the text word by word and when he comes across a verb he introduces two of its word forms: the first one matches the verb lemma and the second one matches the 3rd person masculine of the present tense form. The rest of the methodology can be outlined as follows:

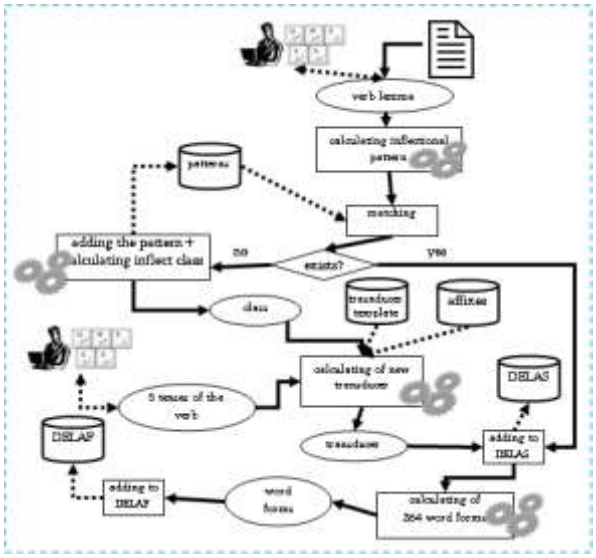

Fig.5. The General Scheme of the Methodology.

1. From the two above mentioned word forms, the inflectional pattern is calculated (cf. Algorithm1 and the relevant subsection),
2. The pattern of the introduced word forms are matched to the ones already stored,
3. If they match this means that the inflectional paradigm (the inflectional finite state transducer or the inflectional graph in the Unitex language) is already known and then we proceed to adding this verb to the lemma dictionary (DELAS) and the word forms dictionary (DELAF)
4. Else the algorithm automatically calculates the new inflectional transducer (inflectional graph) (cf. Algorithm 3 and the relevant subsection)

5.2 The Inflectional Pattern

The inflectional pattern represents the inflectional paradigm in a compact yet transparent notation. The calculation of this compact form relies on four principles:

1. Unlike what is stated in Arabic morphology, the inflection in our algorithm is based on lemma rather than the root
2. As noticed, Arabic inflection affects a well-determined subset of consonants and vowels of the verb lemma at well-determined positions
3. If one can classify Arabic verbs depending on whether or not their consonants and vowels may be affected by the inflection phenomena, then we can find classes represent the different inflection paradigms of the verbs

We can determine the inflection paradigm of a verb from its inflection pattern.

[10] The content of our tag set can be found at the link http://www-igm.univ-mlv.fr/~unitex/zips/Arabic.zip. When unzipping this archive the tag set can be found with the name /Arabic/Dela/tagset.pdf.

[11] The user/annotator should be at least an Arabic native speaker if not a linguist. He manually lemmatizes the verbs of corpus and gives the remaining five forms if needed. The annotation of the corpus can be done offline.

Algorithm 1: Calculating inflectional pattern
Input : verb lemma and 3rd person masculine imperfect active word form
Output : inflection pattern

```
scheme ← ''
for (i=0 to length(lemma)-1) scheme ← scheme + coding(lemma[i], Table1)
scheme ← scheme + ', '
for (i=0 to length(word_form-1) scheme ← scheme + coding(word_form[i], Table1)
return scheme
```

The inflection pattern is calculated as follows:

1. As discussed before, the pattern comprises two parts;
2. The characters of the two word forms are replaced by their corresponding codes, presented in Table 1.

Table 1.Character Correspondence in Inflection Pattern Calculation

ا	ى	و	أ ي	ؤ	ئ ء	آ
A	Y	U	I H	O	W h	M

ت	ن	◌َ	◌ُ	◌ِ	◌ّ	◌ْ	Other wise
t	n	a	u	i	s	o	c

Each of the following examples is presented with the two Arabic word forms of the verb, their HSB transliteration and the English translation of the verb.

Examples:

a) كَتَبَ يَكْتُبُ /kataba yak.tubu/ *to write*
inflection pattern(كَتَبَ يَكْتُبُ)=cacaca cacocucu
b) كَتَبَ يَكْتِبُ /kataba yak.tibu/ *to prescribe*
inflection pattern(كَتَبَ يَكْتِبُ)=cacaca cacocicu
c) وَلَّى يُوَلِّي /wal~a ý yuwal~iy/ *to crown*
inflection pattern(وَلَّى يُوَلِّي)=cacsaY cucacsiI
d) أَدَّى يُؤَدِّي /Ad~aý yuŵd~iy/ *to lead*
inflection pattern(أَدَّى يُؤَدِّي)=HcsaY cuOcsiI

5.3 The New Transducer Generating Algorithm

In the case where the verb inflection transducer is not found, the algorithm proceeds to the calculation of a new transducer by combining the three components detailed in the sections below. These components are the template graph representing the framing of the new transducer (cf. subsection below), the second component, affixes, takes part in building the content of the new transducer boxes[12]. The numerical template is the final form of the transducer boxes (cf. subsection below)

[12] In Unitex terminology, the box is equivalent to a sequence of transducer transitions, it contains the sequence input letters; this sequence is to be recognized. Under the box there is another sequence prefixed by ":" which is to be produced if the recognition succeeds, see the left side of Fig. 6. For further reading the readers are referred to [42].

Algorithm 2: Calculating inflectional class
Input : verb lemma and 3rd person masculine imperfect active word form
Output : affixes

```
switch (lastCharacter(lemma), lastCharacter(word_form) )
    case 'ى', 'ي': { affixes ← affixes2}
    case 'ّ', 'ّ': { affixes ← affixes3}
    case 'ا', 'و': { affixes ← affixes4}
    case 'ى', 'ى': { affixes ← affixes5}
    default : { affixes ← affixes1}
end switch
return affixes
```

5.4 The Template Transducer

It represents a void structure which can be turned into an inflection transducer by filling in the inputs of its boxes and leaving the outputs as they are. An inflection transducer is a set of boxes (cf. Fig.5) where each box represents a full path in the transducer. The number of word forms for an Arabic verb is about 264 (in max case) represented as 184 Unitex graph boxes. As shown in Fig.6 the input and the output of a box are separated by a slash. For instance, <1ت23ّ/:A1smc:A1sfc> is the first box of Fig.6. In the case of the template transducer the input contains a special character * which means *void*. The * will be replaced by the numerical template of the word form in the new generated transducer.

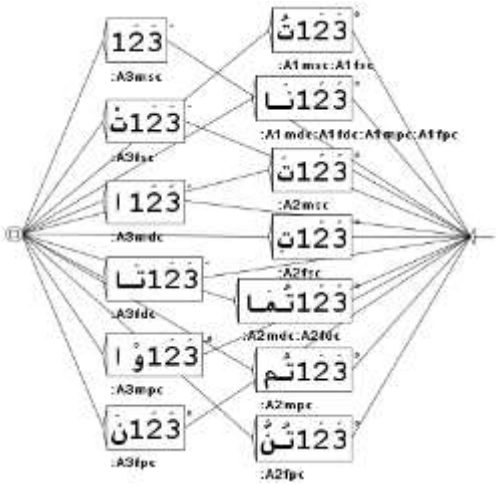

Fig.6. This Figure Shows a Unitex Inflection Transducer Producing 18 Word Forms. The Left Side Shows a Right to Left Graph of Arabic Compliant Transducer. The Right Side is the Equivalent Text Format of the Above Graphic Transducer. Some of the Word Forms are Ambiguous: for Instance, the Second Upper Right Box And Its Textual Equivalent (the 2nd line) Represent One Word Form for Four Different DELAF Entries.

5.5 Affixes

In Arabic, the verb inflection phenomenon affects the prefix, the suffix and sometimes the stem. In the 264 word forms of the verb, the stem takes at most five different forms. In our algorithm, these forms are drawn from the five elements introduced by the user. The affix file contains the prefixes and the suffixes which will be added to each stem in order to construct the numerical template of the word form which will replace the * in the template transducer. Up to that level of progress (70% of

Arabic verbs) we needed only five different classes of affixes; in our algorithm, the choice of the right file of affixes is determined by the Algorithm 2.

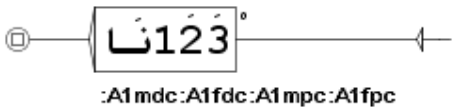

Fig.7. Ambiguity in Unitex Graphs

The core of the affixes file consists of 184 lines. Each line has the structure <*i1*, *i2*, *substr1*, *i3*, *substr2*> and has meaning as follows:

i1: the line number which also corresponds to the box number in the template transducer, it takes values from 0 to 183.

i2: the substring length which is removed at the beginning in the numerical template according to one of the five forms introduced by the user,

substr1: the substring which replaces the removed characters at the beginning of the numerical template,

i3: the substring length which is removed at the end of the numerical template,

substr2: the substring which replaces the removed characters at the end of the numerical template.

Algorithm 3: Calculating the new transducer
Input : transducer template, affixes, form1, form2, form3, form4, form5
Output : new transducer

```
for (i=0 to 183)
    read(line1, transducer_template)   /*the file of transducer template or affixes each of them
    read(line2, affixes)               contains 184 lines*/
    switch (i)
        ≤ 0 and ≥ 12 : form ← form1
        ≤ 13 and ≥ 86 : form ← form2
        ≤ 87 and ≥ 96 : form ← form3
        ≤ 97 and ≥ 109 : form ← form4
        ≤ 110 and ≥ 183 : form ← form5
    end switch
    if (weakAjwaf(form1))
        if (originAlif(form1)='و')
        processAjwaf1(form)   /* for instance قال-يقول */
        if (originAlif(form1)='ي')
        processAjwaf2(form)   /* for instance باع-يبيع */
    endif
    stem ← stemmer(form, line2) /*the stem changes e.g. stem(قُلْتُ)=قُلْ stem(قَالَتْ)=قَالَ */
    prefix ← calculatePrefix(line2)
    suffix ← calculateSuffix(line2)
    form ← morphotaticsAdjusment(prefix+stem+suffix)
    /*the morphotactics rules are

    applied e.g. أَأْخُذُ becomes آخُذُ*/
    numeric_form ← unitexNumericTemplate(form, form1) /*يَقُولُونَ becomes ي1و3ُونَ*/
    line ← constructTransducerLine(line1, numeric_form) /*replace the * by numeric_form in the line of transducer template*/
    add(line, new transducer)
endfor
```

5.6 The Numerical Template

In Unitex, the inflection of Semitic languages is expressed as numerical templates. For instance, Fig. 7 shows the numerical template for 3rd masculine singular of the accusative present tense of a lemma of three consonants. In fact, words are inflected according to consonant skeletons. A lemma is made of consonants, and the inflection process is supposed to enrich this skeleton with short vowels and affixes.

The DELAS entry of the lemma *ktb* in the case of a Semitic language is supposed to be:

ktb,$V31-123

The $ sign before the grammatical code indicates that this is a Semitic entry, and the lemma is the consonant skeleton. The V31-123 indicates the used inflection transducer.

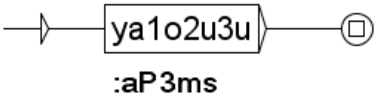

Fig.8. Graph Box Containing a Numerical Template for One DELAF entry

When applying the lemma on the numerical template, the DELAF entry will be:

yakotubu, ktb.V:aP3ms

In our algorithm, the numerical template is calculated from the five word forms introduced by the user. BAMA uses the same five possible stems to produce all possible word forms (more than 260). After removing characters and/or adding affixes as it is explained in section 5.5., we morphologically adjust the result word and then replace the consonant with their numeric order in the lemma. The adjustment is due to orthographic variations due to morphophonemic alternations. Variations are the discrepancies between the underlying or morphophonemic strings and their surface realization, which are either phonological or orthographical strings depending on the purpose of the grammar [24].

Example:

If the user introduces the following 5 word forms: كَتَبَ/kataba/*he wrote*, يَكْتُبُ/yak.tubu/*he writes*, كُتِبَ/kutiba/*is written*, يُكْتَبُ/yuk.tabu/*being written*, أُكْتُبْ/Auk.tub./*you write* (imperative)

- When removing all diacritics from the first word the result is the lemma كتب, then the numeric order is as follows: ك=1, ت=2, ب=3.
- If we try to calculate the numerical template of the 24th box of the inflection transducer of the inflection pattern of example a) in section 5.2 the 24th line in the affixes file will be applied to the stem.
- In this case the stem in question corresponds to the

second word يَكْتُبُ and the affixes line will be: 23,1,ت,1,ـَانِ

- The result sequence will be تَكْتُبَانِ/tak.tubaAni/*they write* (dual) and the corresponding numerical template will be تَ1ـ2ـ3ـانِ/ta1.2u3aAni/.
- The adjustment is not necessary in this case but it is indeed necessary in other cases such as for the lemma صمت/Samata/*be silent* and the 3rd affixes line: 2,-,-,1,ـْتَ
- When applying the affixation on the corresponding stem of the lemma, the result sequence is صَمَتْتَ/Samat.ta/. So it is orthographically not permissible in Arabic to spell the substring تْتَ/t.ta/, it is therefore altered to a geminated تّ/ta/ and the adjusted sequence will be صَمَتَّ/Samat~a/
- The third consonant of the lemma is removed and replaced by the Arabic gemination mark. In this case the numerical template will be تَ1ـَ2ـَتَّ/1a2at~a/instead of تَ1ـْ2ـْ3ـْتَ/1a2a3.ta/.

Algorithm 4: Inflection
Input : inflection transducer, DELAS
Output : minimal transducer of DELAF

```
DELAF ← unitexInflectAPI(DELAS, transducer)
pvDELAF ← calculatePartialyVoweledForms(DELAF)
uvDELAF ← calculateUnvoweledForms(DELAF)
add(pvDELAF, DELAF)
add(uvDELAF, DELAF)
eDic ← unitexLexicographicTreeAPI(DELAF)
eDic ← unitexDeterminizationAPI(eDic)
eDic ← unitexMinimizationAPI(eDic)
return eDic
```

Finally, the acyclic deterministic transducer representing our verb dictionary is constructed; it's made up of two transducers one for the fully voweled word forms and the other for partially and unvoweled word forms. The determinization and minimization allowed us, for instance, to get a transducer with 67,723 states and 235,117 transitions from a dictionary of 6,633,299 entries and 210,791,741 characters (the partially and unvoweled forms dictionary).

VI. Experiment and Results

In this section we expose the outcomes of our approach. Our experiments exploit a corpus which is representative of MSA containing over 5 million words[13]. The table 2 shows the findings of the methodology. If one focuses his attention on the fourth column (verbs/transducer) he can easily notice that the table is divided in two parts. First part consisting of the first four lines and is characterized by the low ratio verbs/transducer. For this part the ratio doesn't exceed 10 verbs per one transducer while it suddenly increases to 64.52 in the fifth line and 78.57 in the last line.

Another relevant point is that the 120 new transducers are created only for this stage which represents 70.17% of all transducers and that only for 1000 verbs (5.93% of all verbs). This finding clearly shows that the great number of inflection paradigms of our corpus verbs is generated in the first stage and means that our inflection system goes to stabilization, i.e. adding new verbs may not need designing new transducers. The last column of Table 2 shows that the growth rate of the new transducers is decreasing from 100% to 9.36% and the curve in Fig. 9 shows that the transducers increase much less than the verbs.

Table 2. Statistics on Obtained Results

Verbs in group	Trans-ducers	New transducers	Verbs/ Transducer	Growth of transducers
100	33	33	3.03	100.00%
200	53	20	3.77	37.73%
500	94	41	5.32	43.61%
1,000	120	26	8.33	21.67%
10,000	155	35	64.52	22.58%
16,855	171	16	78.57	9.36%

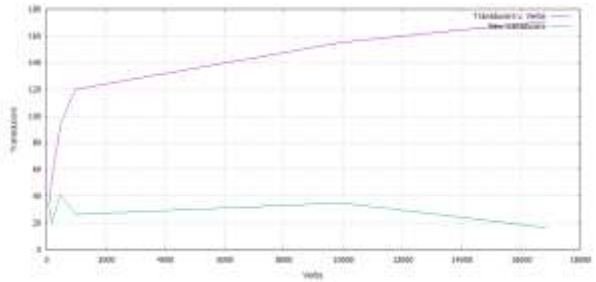

Fig.9. Transducers Increase Compared to Verbs

The experiments also show that the resulting dictionaries have a good coverage. These resources cover more than 70% of Arabic.

Table 3. The Current Content of DELAS and DELAF Dictionaries

POS	Number of DELAS entries	Number of DELAF entries with diacritics	Number of DELAF entries without diacritics	Coverage
Incomplete verbs	13	768	902	100%
Verbs	16,842	4,446,288	6,632,397	+70%
Total	16,855	4,447,056	6,633,299	

Table 3 summarizes the content of the dictionaries; these resources contain 16,855 verb lemmas and 11,080,355 fully, partially and not vocalized verbal word forms. The number of semi-automatically generated transducers is 171. All these resources are publicly

[13] The corpus is freely available at http://aracorpus.e3rab.com/argistestsrv.nmsu.edu/AraCorpus.tar.gz (last accessed Nov 2014). In our experiments we used only a part of this corpus: the original size of the aracorpus is more than 113 millions words [43] however in our experiments we used just a part containing 5 millions words.

available and currently used as an open package in the Unitex framework[14] under the LGPL license.

6.1 Evaluation

In this subsection we try to highlight the contribution, limitations and potential of our approach and its outcome: the content of the dictionaries. Therefore the evaluation of our system is done versus the ALESCO Sarf system. There are several reasons for which we have chosen the Sarf system as our gold reference. First, as it is stated in [15], the Sarf system is built on a strong linguistic basis. Secondly, it is a freely open source system (data and code)[15] for Arabic derivation and inflection. Finally as far as we know it is the best Arabic verb inflection system in terms of coverage percentage.

In the evaluation process, we sampled randomly 100 verbs from the 16,855 verbs of our resources and we tried to compare the output of their inflection by our system to the inflection output of the gold reference. In this evaluation we used Sarf system as a verb conjugator.

The standard evaluation metrics namely precision (*P*), recall (*R*) and f-measure are calculated upon the True Positive (*TP*), False Positive (*FP*) and False Negative (*FN*) values as follows:

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

$$Fmeasure = \frac{2\,P\,R}{P + R}$$

In the case of our system the *TP* represents the number of the correct produced word forms and also validated by the reference system. The *TP* equals 21,247 word forms. The *FP* in turn represents the wrong word forms of our system, it equals 4,953 word forms and finally *FN* represents the number of word forms that are not produced by our system but produced by the reference system. Then the evaluation metrics are as follows:

$$Precision = \frac{21247}{21247 + 4953} = 81.10\%$$

$$Recall = \frac{21247}{21247 + 42} = 99.80\%$$

$$Fmesure = \frac{2 \times 0.81 \times 0.99}{0.81 + 0.99} = 89.48\%$$

As we can notice our system has high *Recall* and reasonable *Precision*; the low *Precision* compared to the *Recall* is due to the overgeneration problem of our system: the surplus word forms produced by the surplus paths of the transducers of our system. These surplus paths correspond to the morphological features number and gender of the 1st person. In MSA there are only two true possibilities (singular and plural for neuter gender) however in our transducer there are a total of six paths i.e. four extra paths and thus producing four extra wrong word forms. This problem can be simply corrected by deleting these extra paths in the 171 transducers.

[14] The open dictionaries of this work can be found at http://www-igm.univ-mlv.fr/~unitex/zips/Arabic.zip (last accessed Dec 2014)

[15] The packages and documentation of the Sarf system can be found at http://sourceforge.net/projects/sarf/ (last accessed Dec 2014)

VII. Conclusions and Perspectives

Although the traditional grammar of the Arabic morphology and the editorial dictionaries are root-pattern based, we can conclude that building Arabic lexica based on lemma is more suitable for NLP tasks and easy to achieve. The experimentation and the evaluation of our system against the gold reference allowed us to conclude that our system is readily adaptable not only for verbs but also for nouns aside their types; however the same experimentation have shown that Sarf system produces word forms only for nouns that derived from the triliteral and quadriliteral roots. We consider the contribution in the current paper as a first step in building an Arabic lexica (verbs) and we plan the building of an analog lexica for Arabic nominals (nouns, adjectives and gerunds) in future research. All built resources will be distributed as open packages in the Unitex platform in the hope to be used and deployed in various Arabic NLP tasks such as Arabic named entity recognition and classification among others.

Acknowledgment

We would like to thank Ali Rahmouni and the members of the LMMC laboratory at the University of Sa ïda for their moral and material support; they made under our disposal all physical means of their laboratory during the completion of the work and writing this paper.

We would like to thank also some colleagues of the University of Tours: the members of the BdTln research team, especially Dr Emeline Lecuit of the LLL linguistic laboratory for her inputs and comments for this paper.

References

[1] N. Doumi, A. Lehireche, D. Maurel and M. Ali Cherif, “La conception d'un jeu de ressources libres pour le TAL arabe sous Unitex,” in TRADETAL2013, Colloque international en Traductologie et TAL, Oran - Algeria, 2013.

[2] S. Khoja, "APT: Arabic Part-of-Speech Tagger." in *Proceedings of the Student Workshop at the 2nd Meeting of the NAACL, (NAACL'01)*. 2001. Carnegie Mellon University, Pennsylvania. pp. 20-25.

[3] M. Attia, P. Pecina, A. Toral, L. Tounsi, J. Van Genabith, "A lexical database for modern standard Arabic interoperable with a finite state morphological transducer." in *Procedding of Second International Workshop, SFCM Systems and Frameworks for Computational Morphology*. 2011. Zurich, Switzerland,: Springer. pp. 98-118.

[4] N. Habash, *Introduction to Arabic natural language processing*: Synthesis lectures on human language technologies, Morgan & Claypool, 2010.

[5] D. Maurel, and F. Guenthner, *Automata and dictionaries*, Texts in computing, ed. I. Mackie. Vol. 6. London: King's

college, 2005.
[6] L. Clément, B. Lang, and B. Sagot, "Morphology based automatic acquisition of large-coverage lexica." in *LREC04 4th International Conference on Language Resources and Evaluation*. 2004. Lisbon, Portugal. pp. 1841-1844.
[7] I. M. H. Saleh, and N. Habash, "Automatic extraction of lemma-based bilingual dictionaries for morphologically rich languages," in Third Workshop on Computational Approaches to Arabic Script-based Languages at the MT Summit XII, Ottawa, Canada, 2009.
[8] N. Habash, and O. Rambow, "Arabic tokenization, part-of-speech tagging and morphological disambiguation in one fell swoop." in *Proceedings of the 43rd Annual Meeting of ACL*. 2005. Ann Arbor, Michigan. pp. 573-580.
[9] F. J. Och, and H. Ney, "A systematic comparison of various statistical alignment models," *Computational Linguistics*, vol. 29, no. 1, pp. c-51, 2003.
[10] P. Koehn, "Pharaoh: A beam search decoder for phrase-based statistical machine translation models," *Machine Translation: From Real Users to Research, Proceedings*, Lecture Notes in Computer Science R. E. Frederking and K. B. Taylor, eds., pp. 115-124, 2004.
[11] W. W. Cohen, "Learning trees and rules with set-valued features." in *13th National Conference on Artificial Intelligence (AAAI 96) / 8th Conference on Innovative Applications of Artificial Intelligence (IAAI 96)*. 1996. Portland. pp. 709-716.
[12] M. Dreyer, and J. Eisner, "Discovering morphological paradigms from plain text using a Dirichlet process mixture model." in *Proceedings of the Conference on Empirical Methods in Natural Language Processing*. 2011. Association for Computational Linguistics. pp. 616-627.
[13] G. Durrett, and J. DeNero, "Supervised Learning of Complete Morphological Paradigms." in *The 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACLHLT)*. 2013. Atlanta. pp. 1185-1195.
[14] T. Buckwalter, "Buckwalter Arabic Morphological Analyzer Version 2.0," *catalog number LDC2004L02*, LDC, 2004.
[15] A. A. Neme, "A lexicon of Arabic verbs constructed on the basis of Semitic taxonomy and using finite-state transducers." in *International Workshop on Lexical Resources*. 2011. Slovenia. pp. 78-85.
[16] D. Graff, M. Maamouri, B. Bouziri *et al.*, "Standard arabic morphological analyzer (SAMA) version 3.1," *Linguistic Data Consortium LDC2009E73*, 2009.
[17] R. Abbes, J. Dichy, and M. Hassoun, "The architecture of a standard Arabic lexical database: some figures, ratios and categories from the DIINAR.1 source program." in *Proceedings of the Workshop on Computational Approaches to Arabic Script-based Languages*. 2004. Geneva, Switzerland: Association for Computational Linguistics. pp. 15-22.
[18] O. Smrz, "ElixirFM - Implementation of Functional Arabic Morphology." in *ACL2007, Computational Approaches to Semitic Languages: Common Issues and Resources*. 2007. Prague, Czech Republic. pp. 1-8.
[19] M. Al-Bawab, *Arabic derivation and inflection algorithms*, ALESCO : Arab League Educational, Scientific and Cultural Organization, Tunisia, 2007.
[20] M. Silberztein, "NooJ: an oriented object approach." in *INTEX pour la Linguistique et le Traitement Automatique des Langues, Actes des 4èmes et 5èmes journées INTEX, Bordeaux, may 2001 and Marseille, may 2002*. 2004. Besançon: Presses universitaires de Franche-Comté
[21] S. Mesfar, "Analyse morpho-syntaxique automatique et reconnaissance des entités nommées en arabe standard," PhD thesis, Université de Franche-Comté, 2008.
[22] E. F. Moore, "Gedanken-experiments on sequential machines," *Automata studies*, vol. 34, pp. 129-153, 1956.
[23] M. Mohri, "On some applications of finite-state automata theory to natural language processing," *Natural Language Engineering*, vol. 2, no. 1, pp. 61-80, 1996.
[24] K. R. Beesley, "Arabic morphology using only finite-state operations." in *Proceedings of the Workshop on Computational Approaches to Semitic languages*. 1998. Association for Computational Linguistics. pp. 50-57.
[25] M. Mohri, "Compact representations by finite-state transducers." in *Proceedings of the 32nd annual meeting on Association for Computational Linguistics*. 1994. Association for Computational Linguistics. pp. 204-209.
[26] J. Daciuk, "Incremental construction of finite-state automata and transducers, and their use in the natural language processing," PhD thesis, Technical University of Gdańsk, 1998.
[27] J. Daciuk, B. W. Watson, and R. E. Watson, "Incremental construction of minimal acyclic finite state automata and transducers." in *Proceedings of the International Workshop on Finite State Methods in Natural Language Processing*. 1998. Association for Computational Linguistics. pp. 48-56.
[28] J. Daciuk, S. Mihov, B.W. Watson, R.E. Watson, "Incremental construction of minimal acyclic finite-state automata," *Computational Linguistics*, vol. 26, no. 1, pp. 3-16, Mar, 2000.
[29] M. Mohri, F. C. N. Pereira, and M. D. Riley, "Systems and methods for determinization and minimization a finite state transducer for speech recognition," Google Patents, 2001.
[30] J. Daciuk, "Comparison of construction algorithms for minimal, acyclic, deterministic, finite-state automata from sets of strings," *Implementation and Application of Automata*, pp. 255-261: Springer, 2003.
[31] B. W. Watson, and J. Daciuk, "An efficient incremental DFA minimization algorithm," *Natural Language Engineering*, vol. 9, no. 1, pp. 49-64, 2003.
[32] R. C. Carrasco, J. Daciuk, and M. L. Forcada, "An implementation of deterministic tree automata minimization," *Implementation and Application of Automata*, pp. 122-129: Springer, 2007.
[33] R. C. Carrasco, J. Daciuk, and M. L. Forcada, "Incremental construction of minimal tree automata," *Algorithmica*, vol. 55, no. 1, pp. 95-110, 2009.
[34] L. Tounsi, B. Bouchou, and D. Maurel, "A compression method for natural language automata." in *Proceeding of the 2009 conference on Finite-State Methods and Natural Language Processing: Post-proceedings of the 7th International Workshop FSMNLP*. 2009. pp. 146-157.
[35] B. W. Watson, "A taxonomy of algorithms for constructing minimal acyclic deterministic finite automata," *South African Computer Journal*, no. 27, pp. 12-17, August, 2001.
[36] S. Mihov, "Direct construction of minimal acyclic finite states automata," *Annuaire de l'Universite de Sofia St. Kl. Ohridski, Faculté de mathématiques et Informatique*, vol. 92, no. 2, 1999.
[37] K. R. Beesley, and L. Karttunen, "Finite-state non-concatenative morphotactics." in *Proceedings of the 38th Annual Meeting on Association for Computational Linguistics*. 2000. Association for Computational Linguistics. pp. 191-198.

[38] M. Al-Bawab, M. Merayati, Y. Mir Alam, M.H. Al-Tayene, *Statistics on Arabic verbs in the computational lexicon*, Lebanon: Librairie Du Liban Publishers, 1996.
[39] D. E. Kouloughli, *Grammaire de l'arabe d'aujourd'hui*: Pocket, 1994.
[40] B. Courtois, and M. Silberztein, "Dictionnaires électroniques du français," *Langue française*, vol. 87, no. 1, pp. 3-4, 1990.
[41] B. Courtois, *Buts et méthodes de l'élaboration des dictionnaires électroniques du LADL*, Université Paris 7 Denis Diderot: Centre Interlangue d'Études en Lexicologie, 1994-1995.
[42] S. Paumier, *Unitex manual for version 3.1*, http://www-igm.univ-mlv.fr/~unitex/Unitex Manual3.1.pdf: IGM, Université de Marne-la-Vallée, Paris, 2014.
[43] W. Zaghouani, "Critical Survey of the Freely Available Arabic Corpora," in Workshop on Free/Open-Source Arabic Corpora and Corpora Processing Tools - LREC2014, Reykjavik, Iceland, 2014, pp. 1-9.

Authors' Profiles

Noureddine Doumi is currently assistant professor at computer science department in Tahar Moulay University of Saida; he received his Magister degree in computer science from University of Sidi-Bel-Abbes in 2005. He is member of EEDIS lab in UDL-SBA and an active member as a developer in Unitex/GramLab project in University of Paris-Est Marne-La-Vallée. His research interest includes Arabic NLP, Linguistic Resources, Finite-State Machines and Machine Learning.

Ahmed Lehireche has completed respectively ING Diploma from ESI of Algiers (1981) with the final curriculum project at the IMAG (France), "MAGISTER" Diploma from USTOran (1993) and "DOCTORAT D'ETAT" Diploma from UDL Sidi bel Abbes (2005). He is working as a Director of research, head of the Knowledge Engineering Team at the EEDIS laboratory and full Professor at the computer science department of UDL Sidi bel Abbes. He is mainly concerned with AI, Computer Science Theory and Semantics in IT.

Denis Maurel was born in 1956. He received his Ph.D. in computer science from the University Paris 7 (France) in 1989. He has been qualified both in computer science and linguistics. Since 2000, he is Professor at the Université François Rabelais at Tours (France). He is Head of the BdTln (Data base and NLP) team of the LI (Computer Science Laboratory) of this university, member of the steering committee of CIAA conferences. His fields of interest are NLP (with focus in Named entities, Morphology, Resources) and Finite state machines.

Ahmed Abdelali, a Senior Engineer at QCRI, Arabic Language Technologies group, his research interest includes machine translation and Arabic text processing.

He received his PhD degree in computer science from the New Mexico Institute of Mining and Technology, USA in 2006.

How to cite this paper: Noureddine Doumi, Ahmed Lehireche, Denis Maurel, Ahmed Abdelali,"A Semi-Automatic and Low Cost Approach to Build Scalable Lemma-based Lexical Resources for Arabic Verbs", International Journal of Information Technology and Computer Science(IJITCS), Vol.8, No.2, pp.1-13, 2016. DOI: 10.5815/ijitcs.2016.02.01