# The Power of Language Music: Arabic Lemmatization through Patterns

**Mohammed Attia**
Google Inc.
New York City
NY, 10011
attia@google.com

**Ayah Zirikly** and **Mona Diab**
Department of Computer Science
George Washington University
Washington, DC
ayaz,mtdiab@gwu.edu

## Abstract

The interaction between roots and patterns in Arabic has intrigued lexicographers and morphologists for centuries. While roots provide the consonantal building blocks, patterns provide the syllabic vocalic moulds. While roots provide abstract semantic classes, patterns realize these classes in specific instances. In this way both roots and patterns are indispensable for understanding the derivational, morphological and, to some extent, the cognitive aspects of the Arabic language. In this paper we perform lemmatization (a high-level lexical processing) without relying on a lookup dictionary. We use a hybrid approach that consists of a machine learning classifier to predict the lemma pattern for a given stem, and mapping rules to convert stems to their respective lemmas with the vocalization defined by the pattern.

## 1 Introduction

Roots and patterns in Arabic are essential for understanding the derivational aspects of the lexicon. In Arabic, roots and patterns function like meta data. Lemmas or lexical entries recorded in dictionaries only represent a static lexicon at a fixed point in time, while roots and patterns (as part of the mental lexicon) have stronger dynamic role in the creation of new entries and the prediction of their semantic paradigms. So, derivation in Arabic is about the construction of a large semantic forests of concepts that are related through a single grand-parent or a super-lemma, that is the root.

The power of roots and patterns has not yet been fully utilized or understood in Natural Language Processing (NLP). They are traditionally considered as a convenient way for listing words in dictionaries or teaching Arabic for second language learners, but they have a great potential for automatic processing, due to their strong generalizing capacity and their function as an instrument for decomposing word forms. Roots and patterns are the hidden layers through which Arabic speakers organize, memorize and access the Arabic lexicon.

In many NLP tasks, using surface word forms is found to be inefficient as it significantly adds to sparsity, especially in highly inflected languages; thus, some form of normalization is necessary. Normalization in general, and lemmatization in particular, are meant to reduce the variability in word forms by collapsing related words. This has been shown to be beneficial for information retrieval (Larkey et al., 2002; Semmar et al., 2006), parsing (Seddah et al., 2010), summarization (Skorkovská, 2012; El-Shishtawy and El-Ghannam, 2014), document clustering (Korenius et al., 2004), keyphrase extraction (El-Shishtawy and Al-Sammak, 2012), and text indexing and classification (Hammouda and Almarimi, 2010).

From a lexical point of view, normalization can be conducted at the level of the root, stem or lemma. Lemmatization relates surface forms to their canonical base representations (or dictionary lookup form) (Attia and van Genabith, 2013). It is the inverse of inflection (Plisson et al., 2004), as it renders words to a default and uninflected form, or as is the case with Arabic, a least marked form. A lemma is the common denominator (Kamir et al., 2002) of a set of forms that share the same semantic, morphological and syntactic composition, where it represents the least marked word form without any inflectional affixes.

*Proceedings of the Workshop on Cognitive Aspects of the Lexicon*,
pages 40–50, Osaka, Japan, December 11-17 2016.

In Arabic, a verb lemma is chosen to be the perfective, indicative, 3rd person, masculine, and singular such as شَكَرَ $akara[1] "to thank". Whereas a nominal lemma (namely, nouns and adjectives) is in the nominative, singular, masculine (where possible), such as طالِب TAlib "student".

Stemming and lemmatization are quite distinct processes, albeit frequently confused the terms are sometimes used interchangeably (Brits et al., 2005). Stemming strips off prefixes and suffixes leaving a bare stem with no guarantee that the resulting form is a valid standalone word, while lemmatization renders word forms (inflected forms) in their dictionary citation forms. To illustrate this with an example, consider the Arabic verb form ينتظرون 'yanotaZiruwn' "they wait". Stemming will remove the present prefix 'ya' and the plural suffix 'uwn' and leave نتظر 'notaZir' which is a non-word in Arabic. By contrast, full lemmatization will reveal that the word has gone through a morphological alteration process and return the canonical انتظر 'AinotaZar' "to wait" as the base form.

The root, by contrast, is the three (or four) radical based form from which a word is formed, that is نظر nZr for the above example. Kamir et al. (2002) assume that the relationship between a root and a lemma is purely diachronic (related to the historical derivation of words and their semantic net). However, we show that the relationship is not only diachronic, but also synchronic related to inflection, as root radicals remain the pivots for inflectional affixes.

In our approach we treat the lemmatization as a classification problem relying mainly on word patterns. Unlike previous work, we do not use lexicons or morphological rules or analyzers. Our methodology is based on the powerful and instrumental component that patterns play in the Arabic morphology system. For example, verb lemmas are derived from roots selected from 10 morphological patterns and 35 phonological patterns, see Section 3.1. Additionally, verbs are also inflected for the imperfective, passive voice and imperative through patterns. Noun and adjective lemmas are similarly derived either from roots or from verbs through patterns. Nouns are also inflected for the plural (broken plural) selected from a large set of 83 phonological patterns.

This paper shows how the process of derivation is closely tied to a compact list of patterns with a backward and forward movement directions. For the benefit of the research community we make our list of morpho-phonological patterns publicly available for download[2].

### 1.1 Arabic Morphological System

Arabic words are originally formed from roots (triliteral or quadriliteral consonantal base), which are passed through different stages of derivation, inflection, and clitic attachment until they finally appear as surface forms. A root is not a word, as it does not carry vocalization or Part of Speech (POS) category, but it serves as an underlying representation of words, and the pivot on which morphological processes take place. Beside roots, patterns play a fundamental part in Arabic morphology, as they provide the vocalic mold (or scheme) for the root cardinals to be placed.

Patterns are divided into two paradigms: derivational and inflectional. Derivational patterns are responsible for the choice of syntactic (POS) and semantic structures, and they produce dictionary entries. Inflectional patterns are the ones that express morpho-syntactic features (such as gender, number, tense, mood, voice, etc.), i.e. creating variations within the same dictionary entry. For example درس drs is a root for the semantic net relating to studying/teaching; دَرَسَ darasa is a verb "to study" following the pattern R$_1$aR$_2$aR$_3$a, تَدْرُسُ tadorusu is an inflected verb "she studies" following the inflection pattern taR$_1$oR$_2$uR$_3$u, and so on. The root can be considered as the super-lemma relating words within the same semantic field (Kamir et al., 2002), while a lemma is realized by furnishing the root with a POS and derivational pattern, and the word form (or surface form) is realized by applying the inflectional patterns and attaching clitics.

Figure 1 shows the two layers and six tiers involved in the composition of the Arabic morphological system. The derivation layer is non-concatenative and opaque in the sense that it is a level of abstraction

[1] We use the Buckwalter Arabic Transliteration system (`http://www.qamus.org/transliteration.htm`).

[2] `https://sourceforge.net/projects/arabicpatterns/`

| | | | | | |
|---|---|---|---|---|---|
| Derivation | Root | درس drs | | | |
| | POS | V | | | N |
| | Pattern | $R_1aR_2aR_3a$ | | | $maR_1R_2aR_3ap$ |
| | Lemma | daras ’study’ | | | madrasap ’school’ |
| Inflection | Pattern | $yaR_1oR_2iR_3$ | $R_1aR_2aR_3$ | $iR_1oR_2iR_3$ | $maR_1AR_2iR_3$ |
| | Inflected word | yadoris’studies’ | daras ’study’ | {idoris ’study!’ | madAris ’schools’ |

Table 1: Root and Pattern Interdigitation

that affects the choice of a POS, and it does not have a direct explicit surface manifestation. By contrast, the inflection layer is more transparent. It applies both concatenative and non-concatenative morphotactics. Non-concatenative morphotactics (or templates) are used to express the imperfective aspect, passive voice and the imperative mood for verbs as well as broken plural forms for nouns. Concatenative morphotactics are used to express number and gender for both verbs and nouns (in the case of dual and sound plurals) and person for verbs.

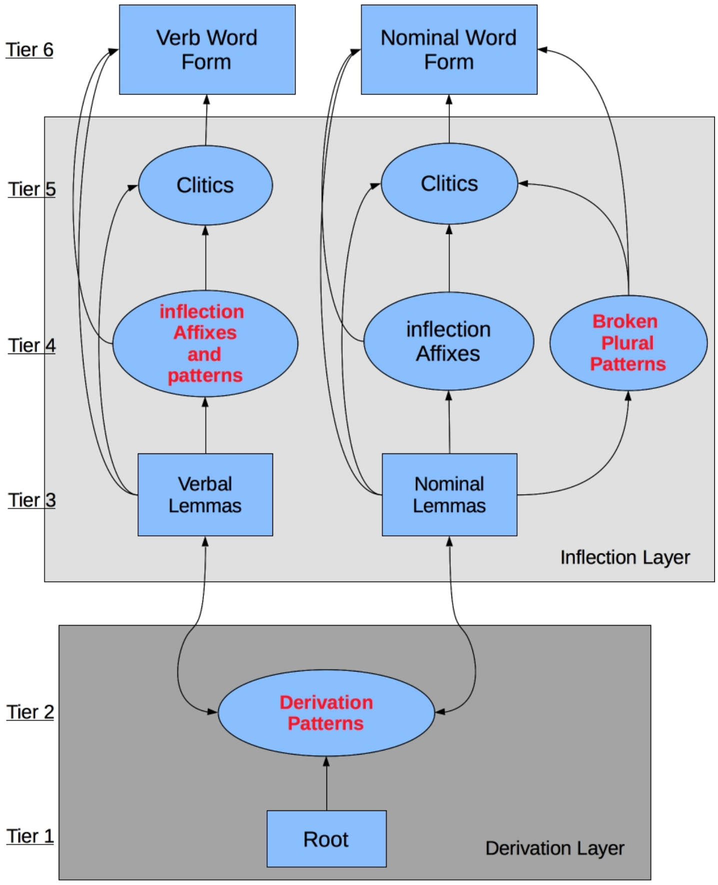

Figure 1: Multi-tier Structure of the Arabic Morphology

A pattern is a conso-vocalic scheme with empty slots, and a root is a sequence of ordered consonants (called radicals), and these radicals are the fillers which occupy the slots according to their linear order. This process of insertion is called interdigitation (Beesley and Karttunen, 2003). An example is shown in Table 1.

To show how the Arabic Lexicon is organized, we can examine the Buckwalter Morphological Analyzer (BAMA) (Buckwalter, 2004) that contains 40,657 entries words, 35,330 (or 87%) of which are derived from 4,494 roots (with an average 7.86 words per root). The remaining words (5,327 or 13%) are not derived and cover function, borrowed and other fixed words.

### 1.2 Vowelization

Arabic short vowels are pronounced in speech, but their representation (Diacritics, or vowel marks) are typically skipped in written text. The lack of diacritization in modern writing is the source of significant ambiguity. For example, the word علم "Elm" can be either عَلِمَ Ealima "to know", عَلَّمَ Eal~ama "to teach", عُلِمَ Eulima "be known", عُلِّمَ Eul~ima "be taught" or عَلَم Elam "flag". Arabic readers intuitively disambiguate based on context. For natural language processing tasks, this disambiguation is necessary, but at the same time not easily achievable.

The Arabic Treebank (ATB) comes with two versions: non-vowelized and fully vowelized. The Buckwalter Morphological Analyzer (BAMA) (Buckwalter, 2004) also provides possible vowelization for words. Together they allow the statistical systems to be trained on a model for vowelization.

Vowelization is an important aspect in the Arabic morphological patterns (which are sometimes referred to as vocalic scheme). Our list of 377 unique patterns is reduced to 175 patterns when vowel marks are removed.

Automatic vowelization, or diacritic restoration, has been discussed in a number of papers. For example, Bebah et al. (2014) describe a hybrid method for automatic vowelization using the Al-Khalil morphological analyzer and a hidden Markov model (HMM) to disambiguate. Some researchers use purely statistical methods for restoring diacritics (Nelken and Shieber, 2005; Elshafei et al., 2006; Ameur et al., 2015; Rashwan et al., 2011).

## 2 Related Work

Lemmatization has been discussed for morphologically rich languages, such as Setswana (Brits et al., 2005), Croatian (Tadić, 2006), Slovene, Serbian, Hungarian, Estonian, Bulgarian and Romanian (in addition to other languages) (Juršič et al., 2007), French (Seddah et al., 2010), Portuguese (da Silva, 2007), Finnish (Korenius et al., 2004), Turkish (Ozturkmenoglu and Alpkocak, 2012) and even English (Balakrishnan and Lloyd-Yemoh, 2014). Plisson et al. (2004) and Juršič et al. (2007) treat lemmatization as a machine learning problem and apply Ripple Down Rule (RDR) induction algorithm to a lexicon of words and their normalized forms to learn lemmatization rules.

Due to the high inflectional nature of the language, it is almost impossible to treat Arabic texts without some sort of normalization. From the implementation point of view, there are basically three approaches for normalizing Arabic: dictionary-based normalization, and statistical normalization, and hybrid normalization.

**Dictionary-based normalization.** The Buckwalter Arabic Morphological Analyser (BAMA) (Buckwalter, 2004) is the most widely used analyser in the literature. The Khoja stemmer (Khoja and Garside, 1999) is a mid-level analyser that falls between a full morphological analyser and a shallow stemmer. It recognizes prefixes and suffixes of a word, and uses patterns to determine the POS tag and extract the root. Hossny et al. (2008) develop an Arabic morphological rule induction system to predict morphological rules using inductive logic programming on sets of example pairs (stem and inflected form) with their feature vectors. El-Shishtawy and El-Ghannam (2012) build a rule-based system that exploits Arabic language knowledge in terms of roots, patterns, affixes, and a set of morpho-syntactic rules to generate lemmas for surface word forms.

**Hybrid normalization.** (Hajič, 2000) argues for the use of a dictionary as a source of morphological analyses for training a statistical POS and morphological tagger for inflectionally rich languages, such as Romanian, Czech, or Hungarian. The method was later applied to Arabic (Hajic et al., 2005). (Roth et al., 2008) develop a system (MADA) that uses statistical methods (SVM classifiers) to perform full morpho-syntactic tagging, along with lemmatization (LexChoice), by selecting the best candidate from the list of competing analyses generated by BAMA (Buckwalter, 2004).

**Statistical normalization.** The Stanford Tagger (Toutanova and Manning, 2000) is a Maximum Entropy POS tagger that has been extended for Arabic, but the problem with this tagger is that it does not perform segmentation of Arabic clitics. AMIRA 2.1 (Diab, 2007; Diab, 2009) uses a supervised SVM-based machine learning method for POS tagging, tokenization, and base phrase chunking. The

|  | Tokens | No alterations | Alterations % |
|---|---|---|---|
| Nouns | 128,294 | 103,363 | 19.43 |
| Verbs | 31,667 | 16,276 | 48.60 |
| Adjectives | 53,177 | 32,599 | 38.70 |
| Proper Nouns | 22,245 | 20,840 | 6.32 |
| Function Words | 65,089 | 59,060 | 9.26 |
| Total | 300,472 | 232,138 | 22.74 |

Table 2: Frequency of alterations in Arabic words

tokenization in AMIRA 2.1 only separates clitics and does not split off inflectional affixes. Abdul-Mageed et al. (2013), develop ASMA, a Memory-Based Learning system that performs fine grained POS tagging and automatic segmentation (stemming) by splitting both inflectional morpheme and clitics, but, like AMIRA, it does not return the lemma of the word.

So far, purely statistical approaches succeeded at developing solutions for normalization at the root and stem levels, but they stopped short of lemmatization. In this paper we introduce the first attempt to treat lemmatization in Arabic as a machine learning classification problem.

## 3 Approach

The use of a machine learning (ML) classifier to directly map words to their lemmas is not feasible in Arabic, due to the fact that Arabic inflection contains change to the internal buildup of the word, as opposed to the straightforward suffixation and prefixation. Table 2 shows the frequency of mismatch between the stem and the lemma in the ATB (Maamouri et al., 2010). We notice that verbs have the highest rate of alterations, or mismatches, (48.6%) followed by adjectives then nouns.

In our work, we use a machine learning classifier to predict the pattern of the lemma for any given surface form (ideally if the words are diacritized and stemmed). In our view the pattern functions as the pivot, or the bridge, between the surface form and the lemma. Our lemmatization is based on two levels of mapping. First, we map the stem to the pattern of the lemma, then we map the pattern of the lemma to the actual lemma form, by extracting the radicals from the stem and filling the slots in the pattern. For training our model, we use the ATB which comes already annotated with lemmas.

Figure 2 shows the architecture of our system. The output to our system is tokenized and POS tagged words, which are then enriched with lemmas and formatted into features that are passed to our machine learning (ML) classifiers. The ML classifiers predict lemmas for given stems, which are then passed into our mapping rules to finally generate the lemmas by merging the stems with the predicted patterns.

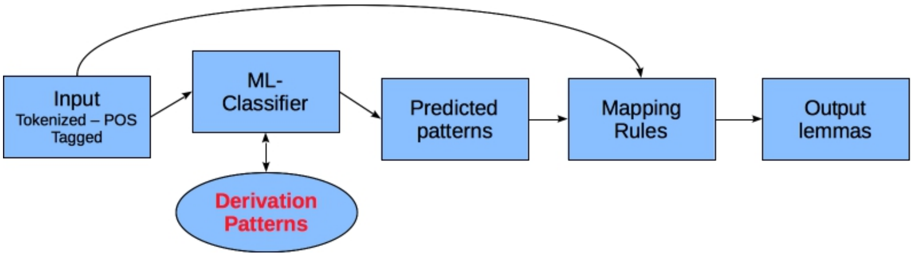

Figure 2: Architecture of the lemmatization system

### 3.1 The Pattern Database

We create a pattern database for all Arabic derivational and broken plural forms. The number of patterns in the database is 655. These are not unique patterns because of the many-to-many relationship between broken plural and singular patterns.

In our work we make a distinction between morphological patterns and phonological patterns. Phonological pattern makes allowance for alterations due to the existence of weak letters, gemination (doubling) and hamzah's (glottal stops). For example, the verb قال qAl "to say" will have the phonological

| Word Type | No. of patterns | Example Pattern | Example word |
|---|---|---|---|
| Broken Plural | 256 (87 unique) | R$_1$aR$_2$A}iR$_3$ | EajA}iz ‘elderly’ |
| Nouns taking broken plural | 135 | R$_1$iR$_2$oR$_3$AR$_4$ | EimolAq ‘giant’ |
| Nouns taking feminine plural | 26 | miR$_1$aR$_2$~ap | miDax~ap ‘pump’ |
| Nouns taking no plural | 20 | taR$_1$oR$_2$AR$_3$ | taokAr ‘remembering’ |
| Active participle اسم الفاعل | 26 | muR$_1$AR$_2$iR$_3$ | muqAtil ‘fighter’ |
| Passive participle اسم المفعول | 22 | muR$_1$aR$_2$~aR$_3$ | muwaj~ah ‘directed’ |
| Verbal nouns المصدر | 42 | {inoR$_1$iR$_2$AR$_3$ | {ino$iTAr ‘spreading’ |
| Nouns of instrument اسم الآلة | 8 | miR$_1$oR$_2$AR$_3$ | mino$Ar ‘saw’ |
| Nouns of Instance اسم المرة | 6 | <iR$_1$AR$_2$ap | <iEAnp ‘assistance’ |
| Adjectives of hyperbole صيغ المبالغة | 11 | R$_1$aR$_2$iyR$_3$ | xabiyr ‘expert’ |
| Attributive adjective الصفة المشبهة | 21 | R$_1$aR$_2$oR$_3$ | Daxom ‘huge’ |
| Verbs | 45 | {iR1otaR2aR3 | {ijotamaE ‘meet’ |
| Names of place اسم المكان | 3 | maR$_1$oR$_2$aR$_3$ | makotab ‘office’ |
| Elative Adjectives اسم التفضيل | 3 | >aR$_1$oR$_2$aR$_3$ | >akoram ‘more generous’ |
| Miscellaneous | 8 | maR$_1$oR$_2$aR$_3$An | mahorajAn ‘carnival’ |

Table 3: Categorization of Arabic patterns

pattern “R$_1$AR$_3$”, while it has the morphological pattern “R$_1$aR$_2$aR$_3$”. Matching via the phonological pattern is computationally more straightforward than matching against the morphological pattern.

The number of unique patterns is 379 (227 are purely morphological patterns, and 152 are purely phonological patterns). Phonological patterns are related to morpho-phonological alteration operations related to the existence of either weak letters or doubling. Weak letters are any of the long vowels (alif, waw, yaa) or hamzah (glottal stop). The traditional Arabic way of representing the root radicals is through the letters f, E, l in their respective order (f = R$_1$, E = R$_2$ and l = R$_3$). Morphological patterns are the representative productive and generic patterns that apply to the majority of words, while phonological patterns are exception sub-branches of the morphological patterns that apply to specific cases where a radical happens to be replaced by a weak letter (long vowel) or hamzah (glottal stop).

In our database we make a fine grained classification of patterns based on their morpho-syntactic functions. Table 3 shows the count of patterns for each type based on POS and plurality paradigm. It is to be noted that in our system proper nouns and foreign Arabized words that do not follow any of the known Arabic patterns and are passed without any further processing.

### 3.2 Dataset and Features

It is hard to directly predict lemmas from stems due to the very fine granularity level. Thus, we generate patterns for all stems and lemmas, which are relatively limited in numbers in comparison to the actual lexical items rendering the search space for the classifier, therefore more manageable. We have two types of patterns: a) automatic patterns generated by replacing all consonants in a word with placeholders, and b) morpho-phonological patterns which only replaces the consonantal base with placeholders. For example, the verb انطلق inoTalaq will be replaced by .i.o.a.a. in the automatic pattern, while it will be replaced by ino.a.a. in the morpho-phonological patterns which correctly identifies the sequence ino outside of the consonantal base. The number of unique automatic lemma patterns is 77, the number of automatic stem patterns is 225, and the number of morpho-phonological lemma patterns is 43 for verbs and 209 for nominals. Then for each stem pattern, we predict the Lemma-pattern (either automatic or morpho-phonological pattern).

The features used in our classifier are the stem, autoStemPattern (the pattern automatically generated from the stem by replacing consonants with placeholders), and affixes (PREF$_0$, PREF$_1$, …, PREF$_n$, SUFF$_0$, SUFF$_1$, …, SUFF$_m$), where n and m are based on the maximum number of prefixes and suffixes, respectively, in the data.

Note that the prefixes and suffixes refer to both clitics (coordinating conjunctions, prepositions and particles) and morphological markers (related to number, gender, person, aspect, mood, etc.), as depicted

| Affixes | Description | Type | List |
|---|---|---|---|
| PREF0 | Conjunctions | pro-clitic | fa, wa |
| PREF1 | Particle | pro-clitic | li, sa, la, mA |
| PREF2 | Perfective marker | prefix | a, >u, na, nu, ta, tu, ya, yu, \| |
| SUFF0 | mood, number, and gender marker | suffix | A, Ani, a, at, atA, aw, awoA, awona, ayo, iy, iyna, nA, na, o, ta, ti, tu, tum, u, uw, uwA, uwna |
| SUFF1 | Accusative pronouns | enclitic | hA, hi, him, himA, hu, hum, humA, ka, ki, kum, kumA, kun∼a, nA, niy |

Table 4: List of affixes for verbs

in Table 4 which shows the affixes for verbs.

Then we compare the performance of two machine learning classifiers to predict the morpho-phonological pattern or the automatic pattern of the lemma for each stem using the features specified above. The results are discussed in Section 4.

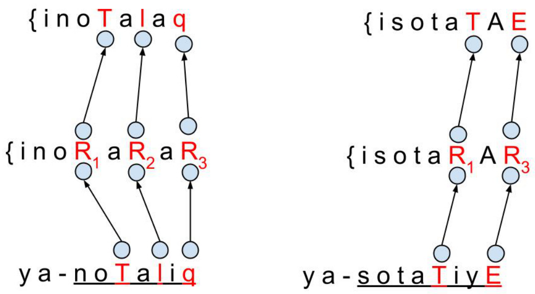

Figure 3: Stem-pattern-lemma Mapping

### 3.3 Mapping from lemma pattern to actual lemma

Using the ML classifier to predict the correct lemma pattern for each stem, we need, given the stem, to map the lemma pattern to the actual realization. We use deterministic rules to map the stem to the lemma using the predicted lemma pattern. We take the radicals from the stem and fill them in place of the pattern's slots in a reverse manner, i.e. starting from the end to the beginning of the string. The rules follow these procedures:

1. Remove prefixes and suffixes from stem
2. Remove diacritics (~, a, i, u, o) from stem
3. Remove weak letters (A, y, w)
4. From end to beginning replace the slot in the pattern with the radical from stem

For example, given the word yanoTaliq "set off" which has the lemma pattern inoR$_1$aR$_2$aR$_3$, first we remove the prefix ya, and the diacritics, that result in nTlq. Then, from the end to the beginning we fill the slots in the pattern with letters from the stem until all slots are consumed, thus replacing R$_3$ with q, R$_2$ with l, and R$_1$ with T. The same goes for yasotaTiyE, except that we additionally ignore the weak letters y and A. Figure 3 shows the mapping process for two words: the first with a morphological pattern (no weak letters), and the second with a phonological pattern (with weak letters).

## 4 Experiments and Evaluation

The method we develop is meant as a proof-of-concept that shows the usability of patterns in the sub-task of retrieving lemmas. It takes as input tokenized and POS-tagged texts. Due to the fact that we are not developing a full scale morphological processor, we cannot compare our results with state-of-the-art applications, such as MADA (Roth et al., 2008), and therefore we use intrinsic evaluation.

We evaluate our approach on the diacritized and undiacritized version of the ATB. For the baseline we consider the stem as the lemma without any further processing. Our data contains 128,293 nouns, 31,666 verbs, and 35,176 adjectives. We divide the data into 80% for training, 10% for development, and 10% for testing. The results in this section are reported against the test set.

Our method consists of two steps. First we use ML classifier to predict the lemma pattern for a given stem. For the ML step we notice that the results in general are remarkably better than the baseline. We conduct two classification experiments. In the first we take the set of morpho-phonological patterns (morphPtrn) as the prediction class, and in the second we use patterns automatically generated from the lemmas by removing cardinal letters (autoPtrn). For example, an autoPtrn can be generated from the lemma انتقال AinotiqAl "moving", which can be transformed into an autoPtrn by replacing all consonants with a placeholder "Ai.o.i.A.". This allows us to automatically generate a list of patterns without being constrained by a manually constructed list.

| **Baseline** | 72.35 | | | |
|---|---|---|---|---|
| **classifier/ template** | **diac** | | **nodiac** | |
| | tmpl_pred | lemma_mapping | tmpl_pred | lemma_mapping |
| Tree_morphPtrn | 98.45 | **98.23** | 94.4 | 89.35 |
| Tree_autoPtrn | 98.67 | 94.12 | 94.34 | **94.03** |
| extraTree_morphPtrn | 97.73 | 92.98 | 90.38 | 77.36 |
| extraTree_autoPtrn | 99.05 | 90.14 | 94.28 | 81.54 |

Table 5: Verbs Results

We tested two ML algorithms: Decision Trees (C4.5) and Extra Trees classifier. We notice that using C4.5 produces significantly better results than Extra Trees. The second step is related to the reconstruction of the actual lemma from combining the stem and the predicted pattern using mapping rules. The results of this step show a marginal loss on the output of the prediction step. The experimental results are shown in Tables 5, 6 , and 7, for verbs, nouns and adjectives respectively, and for both diacritized undiacritized words. The overall results are largely higher than the baseline with diacritized texts. For example, the baseline for verbs is 72.35% while our best result is 98.23%, with similar results for both adjectives and nouns.

With undiacritized words, the performance of the process varies with the type of entries. With verbs and nouns our results are higher than the baseline, with adjectives the prediction scores for patterns remains consistently high (mostly above 95%). But the mapping from pattern to lemma seems to fall below the baseline. Our justification is that adjectives do not undergo as many non-concatenative derivation. Moreover, mapping rules for undiacritized adjectives needs more improvement.

One of the interesting results we found in these experiments is that results with autoPtrn are comparable to (and sometimes even better than) morphPtrn which is an indication that patterns are machine learnable and that we do not need to rely solely on hand-crafted lists of Arabic templates.

| **Baseline** | 81.95 | | | |
|---|---|---|---|---|
| **classifier/ template** | **diac** | | **nodiac** | |
| | tmpl_pred | lemma_mapping | tmpl_pred | lemma_mapping |
| Tree_morphPtrn | 97.93 | **94.6** | 96.48 | **86.59** |
| Tree_autoPtrn | 98.22 | 93.29 | 96.51 | 70.64 |
| extraTree_morphPtrn | 95.66 | 87.2 | 95.59 | 79.2 |
| extraTree_autoPtrn | 96.93 | 87.55 | 92.84 | 63.18 |

Table 6: Nouns Results

| **Baseline** | 93.10 | | | |
|---|---|---|---|---|
| **classifier/** | **diac** | | **nodiac** | |
| **template** | tmpl_pred | lemma_mapping | tmpl_pred | lemma_mapping |
| Tree_morphPtrn | 93.32 | *96.61* | 97.04 | 79.60 |
| Tree_autoPtrn | 99.09 | **97.92** | 96.7 | **86.20** |
| extraTree_morphPtrn | 98.16 | 91.64 | 97.4 | 76.01 |
| extraTree_autoPtrn | 99.81 | 96.13 | 96.47 | 82.61 |

Table 7: Adjectives Results

## 5 Conclusion

We develop successful lemmatization method for Arabic without a dictionary or morphological analyzer. Our approach can serve as a plug-in to stemming applications and POS taggers. It needs to be fed the vowelized (diacritized) stem, the surrounding affixes and the POS tag to be able to return the correct lemma.

Although patterns have occupied center stage in traditional grammar and second language teaching for generations, they have been largely ignored in natural language processing. In this paper we have shown how the complex derivational and inflectional morphological system for Arabic can be modeled by machine learning methods when using patterns as an abstraction level to generalize on the variant surface forms. We also show how the cardinals of the root obey the linear order in various derivations and inflection, making filling the slots in the patterns a straightforward job. This paper describes the first attempt to relate surface forms to their lemmas in Arabic using probabilistic methods.

The recent few years have seen intense interest in deep learning and neural embeddings. In future work we want to handle the same problem with LSTM-based sequence-to-sequence models such as the neural encoder-decoder for morphological re-inflection explained in Kann and Schütze (2016), and test if direct mapping from words to lemmas is feasible, or patterns still represent a necessary component to mediate the process.

## References

Muhammad Abdul-Mageed, Mona T Diab, and Sandra Kübler. 2013. Asma: A system for automatic segmentation and morpho-syntactic disambiguation of modern standard Arabic. In *RANLP*, pages 1–8.

Mohamed Seghir Hadj Ameur, Youcef Moulahoum, and Ahmed Guessoum. 2015. Restoration of Arabic diacritics using a multilevel statistical model. In *Computer Science and Its Applications*, pages 181–192. Springer.

Mohammed Attia and Josef van Genabith. 2013. A jellyfish dictionary for Arabic. In *Electronic lexicography in the 21st century: thinking outside the paper: proceedings of the eLex 2013 conference, 17-19 October 2013, Tallinn, Estonia*, pages 195–212.

Vimala Balakrishnan and Ethel Lloyd-Yemoh. 2014. Stemming and lemmatization: a comparison of retrieval performances. *Lecture Notes on Software Engineering*, 2(3):262.

Mohamed Bebah, Chennoufi Amine, Mazroui Azzeddine, and Lakhouaja Abdelhak. 2014. Hybrid approaches for automatic vowelization of Arabic texts. *arXiv preprint arXiv:1410.2646*.

Karien Brits, Rigardt Pretorius, and Gerhard B van Huyssteen. 2005. Automatic lemmatization in Setswana: Towards a prototype. *South African Journal of African Languages*, 25(1):37–47.

Tim Buckwalter. 2004. Buckwalter Arabic morphological analyzer version 2.0. linguistic data consortium, university of pennsylvania, 2002. ldc cat alog no.: Ldc2004l02. Technical report, ISBN 1-58563-324-0.

João Ricardo Martins Ferreira da Silva. 2007. Shallow processing of Portuguese: From sentence chunking to nominal lemmatization. *Master's thesis*.

Mona Diab. 2007. Improved Arabic base phrase chunking with a new enriched POS tag set. In *In Proceedings of the 2007 Workshop on Computational Approaches to Semitic Languages: Common Issues and Resources*, pages 89–96, Prague, Czech Republic.

Mona Diab. 2009. Second generation AMIRA tools for Arabic processing: Fast and robust tokenization, pos tagging, and base phrase chunking. In *In Proceedings of the Second International Conference on Arabic Language Resources and Tools*, pages 285–288, Cairo, Egypt.

Tarek El-Shishtawy and Abdulwahab Al-Sammak. 2012. Arabic keyphrase extraction using linguistic knowledge and machine learning techniques. *arXiv preprint arXiv:1203.4605*.

Tarek El-Shishtawy and Fatma El-Ghannam. 2012. An accurate Arabic root-based lemmatizer for information retrieval purposes. *arXiv preprint arXiv:1203.3584*.

Tarek El-Shishtawy and Fatma El-Ghannam. 2014. A lemma based evaluator for semitic language text summarization systems. *arXiv preprint arXiv:1403.5596*.

Moustafa Elshafei, Husni Al-Muhtaseb, and Mansour Alghamdi. 2006. Statistical methods for automatic diacritization of Arabic text. In *The Saudi 18th National Computer Conference. Riyadh*, volume 18, pages 301–306.

Jan Hajic, Otakar Smrz, Tim Buckwalter, and Hubert Jin. 2005. Feature-based tagger of approximations of functional Arabic morphology. In *Proceedings of the Workshop on Treebanks and Linguistic Theories (TLT), Barcelona, Spain*.

Jan Hajič. 2000. Morphological tagging: Data vs. dictionaries. In *Proceedings of the 1st North American chapter of the Association for Computational Linguistics conference*, pages 94–101. Association for Computational Linguistics.

Faten Khalfallah Hammouda and Abdelsalam Abdelhamid Almarimi. 2010. Heuristic lemmatization for Arabic texts indexation and classification 1.

Ahmad Hossny, Khaled Shaalan, and Aly Fahmy. 2008. Automatic morphological rule induction for Arabic. In *Proceedings of the LREC'08 workshop on HLT & NLP within the Arabic world: Arabic Language and local languages processing: Status Updates and Prospects*, pages 97–101.

Matjaž Juršič, Igor Mozetič, and Nada Lavrač. 2007. Learning ripple down rules for efficient lemmatization. In *Proceedings of the 10th international multiconference information society, IS*, pages 206–209.

Dror Kamir, Naama Soreq, and Yoni Neeman. 2002. A comprehensive NLP system for modern standard Arabic and modern Hebrew. In *Proceedings of the ACL-02 workshop on Computational approaches to semitic languages*, pages 1–9. Association for Computational Linguistics.

Katharina Kann and Hinrich Schütze. 2016. Single-model encoder-decoder with explicit morphological representation for reinflection. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, Berlin, Germany*, pages 555–560.

Shereen Khoja and Roger Garside. 1999. Stemming Arabic text. *Lancaster, UK, Computing Department, Lancaster University*.

Tuomo Korenius, Jorma Laurikkala, Kalervo Järvelin, and Martti Juhola. 2004. Stemming and lemmatization in the clustering of Finnish text documents. In *Proceedings of the thirteenth ACM international conference on Information and knowledge management*, pages 625–633. ACM.

Leah S Larkey, Lisa Ballesteros, and Margaret E Connell. 2002. Improving stemming for Arabic information retrieval: Light stemming and co-occurrence analysis. In *Proceedings of the 25th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 275–282. ACM.

Rani Nelken and Stuart M Shieber. 2005. Arabic diacritization using weighted finite-state transducers. In *Proceedings of the ACL Workshop on Computational Approaches to Semitic Languages*, pages 79–86. Association for Computational Linguistics.

Okan Ozturkmenoglu and Adil Alpkocak. 2012. Comparison of different lemmatization approaches for information retrieval on Turkish text collection. In *Innovations in Intelligent Systems and Applications (INISTA), 2012 International Symposium on*, pages 1–5. IEEE.

Joël Plisson, Nada Lavrac, Dunja Mladenic, et al. 2004. A rule based approach to word lemmatization. *Proceedings of IS-2004*, pages 83–86.

Mohsen Rashwan, Mohamed Al-Badrashiny, Mohamed Attia, Sherif Abdou, and Ahmed Rafea. 2011. A stochastic Arabic diacritizer based on a hybrid of factorized and unfactorized textual features. *Audio, Speech, and Language Processing, IEEE Transactions on*, 19(1):166–175.

Ryan Roth, Owen Rambow, Nizar Habash, Mona Diab, and Cynthia Rudin. 2008. Arabic morphological tagging, diacritization, and lemmatization using lexeme models and feature ranking. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics on Human Language Technologies: Short Papers*, pages 117–120. Association for Computational Linguistics.

Djamé Seddah, Grzegorz Chrupała, Özlem Çetinoğlu, Josef Van Genabith, and Marie Candito. 2010. Lemmatization and lexicalized statistical parsing of morphologically rich languages: The case of French. In *Proceedings of the NAACL HLT 2010 First Workshop on Statistical Parsing of Morphologically-Rich Languages*, pages 85–93. Association for Computational Linguistics.

Nasredine Semmar, Meriama Laib, and Christian Fluhr. 2006. Using stemming in morphological analysis to improve Arabic information retrieval. In *TALN 2006*, pages 317–326, Leuven, Belgium.

Lucie Skorkovská. 2012. Application of lemmatization and summarization methods in topic identification module for large scale language modeling data filtering. In *Text, Speech and Dialogue*, pages 191–198. Springer.

Marko Tadić. 2006. Croatian lemmatization server. In *Fifth International Conference Formal Approaches to South Slavic and Balkan languages (FASSBL).*

Kristina Toutanova and Christopher D Manning. 2000. Enriching the knowledge sources used in a maximum entropy part-of-speech tagger. In *Proceedings of the 2000 Joint SIGDAT conference on Empirical methods in natural language processing and very large corpora: held in conjunction with the 38th Annual Meeting of the Association for Computational Linguistics-Volume 13*, pages 63–70. Association for Computational Linguistics.