Build Fast and Accurate Lemmatization for Arabic

Hamdy Mubarak
QCRI, Hamad Bin Khalifa University (HBKU), Doha, Qatar
hmubarak@hbku.edu.qa

Abstract

In this paper we describe the complexity of building a lemmatizer for Arabic which has a rich and complex morphology, and show some differences between lemmatization and surface stemming, i.e. removing prefixes and suffixes from words. We discuss the need for a fast and accurate lammatization to enhance Arabic Information Retrieval results. We also introduce a new dataset that can be used to test lemmatization accuracy, and an efficient lemmatization algorithm that outperforms state-of-the-art Arabic lemmatization in terms of accuracy and speed. We share the dataset and the code for research purposes.

Keywords: Arabic NLP, Lemmatization, Stemming, Information Retrieval, Diactitization

1. Introduction

Lemmatization is the process of finding the base form (or lemma) of a word by considering its inflected forms. Lemma is also called dictionary form, or citation form, and it refers to all words having the same meaning.

Lemmatization is an important preprocessing step for many applications of text mining and question-answering systems. Researches in Arabic Information Retrieval (IR) systems show the need for representing Arabic words at lemma level for many applications, including keyphrase extraction (El-Shishtawy and Al-Sammak, 2009) and Machine Translation (Dichy and Fargaly, 2003). In addition, lemmatization provides a productive way to generate generic keywords for search engines (SE) or labels for concept maps (Plisson et al., 2004).

Word stem is that core part of the word that never changes even with morphological inflections; the part that remains after prefix and suffix removal. Sometimes the stem of the word is different than its lemma, for example the words: believe, believed, believing, and unbelievable share the stem (believ-), and have the normalized word form (believe) standing for the infinitive of the verb (believe).

While stemming tries to remove prefixes and suffixes from words that appear with inflections in free text, lemmatization tries to replace word suffixes with (typically) different suffix to get its lemma. For languages having complex derivational and inflectional morphology, like Arabic, lemmatization needs more than just suffix replacement as will be described in next section.

This paper is organized as follows: Section 2. gives some background about Arabic morphology and shows some complexities in building Arabic lemmatization; Section 3. lists IR clustering methods and gives examples to show that lemmatization can enhance search results; Section 4. surveys prior work on Arabic stemming and lemmatization; Section 5. introduces the dataset that we created to test lemmatization accuracy; Section 6. describes the algorithm of the system that we built. Results and error analysis are described in section 7.; and Section 8. concludes the paper and lists some tasks for future work.

2. Background

Arabic is the largest Semitic language spoken by almost 300 million people. It's one of the six official languages in the United Nations, and the fifth most widely spoken language after Chinese, Spanish, English, and Hindi[1].

Arabic has a very rich morphology, both derivational and inflectional. Generally, Arabic words are derived from a root that uses three or more consonants to define a broad meaning or concept, and they follow some templatic morphological patterns (الموازين الصرفية). By adding vowels, prefixes and suffixes to the root, word inflections are generated. For instance, the word وسيفتحونها (wsyftHwnhA)[2] "and they will open it" has the triliteral root فتح (ftH), which has the basic meaning of opening, has prefixes و+س (w+s) "and+will", suffixes ون +ها (wn+hA) "they..it", stem يفتح (yftH) "open", and lemma فتح (ftH) "the concept of opening".

Arabic verbs have the following grammatical categories: **tense** (past, present, imperative, and future), **number** (singular, dual, and plural), **person** (first, second, and third), **mood** (indicative, subjunctive, and jussive for present verbs, given for past verbs, and jussive for imperative verbs), **gender** (masculine and feminine) and **voice** (active and passive).
Typically, lemmatization of a verb is achieved by obtaining its past tense without any prefixes or suffixes, singular number, third person, given mood, masculine gender, and active voice. Mapping between different grammatical values cannot be done in many cases by just stripping word from its prefixes and suffixes but by applying some complex morphological rules due to the derivational nature of Arabic morphology. Table 1 shows some examples.

Arabic nouns and adjectives have the following grammatical categories: **case** (nominative, accusative, and genitive), **number** (singular, dual, and plural (proper or broken plu-

[1] https://en.wikipedia.org/wiki/Arabic

[2] Words are written in Arabic, transliterated using Buckwalter transliteration, and translated.

Case	Example
present->past	قال <- يقول (yqwl, qAl) "said, say"
passive->active	ناقش <- نوقش (nwq$, nAq$) "was discussed, discussed"
first->third	نام <- نمت (nmt, nAm) "I slept, he slept"
plural->singular	رضي <- رضوا (rDwA, rDy) "they satisfy, he satisfies"

Table 1: Examples of complex verb lemmatization cases

rals جمع المذكر والمؤنث السالم والتكسير)), **gender** (masculine and feminine) and **definiteness** (definite and indefinite). Typically, lemmatization of a noun or an adjective is achieved by obtaining its nominative case without any prefixes or suffixes, singular number, masculine gender, and indefinite form. Mapping between different values is not straightforward in many cases as shown in Table 2.

Case	Example
broken plural->singular	رجال <- رجل (rjAl, rjl) "men, man"
proper plural->singular	سنوات <- سنة (snwAt, snp) "years, year"
feminine->masculine	أخضر <- خضراء (xDrA', >xDr) "green (f), green (m)"
genitive->nominative	بناء <- بنائه (bnA}h, bnA') "building-it, building"
special cases	مستشفى <- مستشفيات (mst$fyAt, mst$fY) "hospitals, hospital"
	فيديو <- فيديوهات (fydywhAt, fydyw) "videos, video"...

Table 2: Examples of complex noun lemmatization cases

In addition, according to Arabic morphology and writing system, attaching pronouns to words in some cases changes their last letter. This adds an extra complexity when obtaining lemmas. For example, when nouns ending with Taa-Marbouta letter are attached to possessive pronouns, it will be changed to Taa letter as in حضارة+ه <- حضارته (hDArp+h, HDArth) "its civilization". Also, when verbs ending with Alif-Maqsoura letter are attached to some subject pronouns, it will be changed to Yaa letter as in هدى +نا <- هدينا (hdY+nA, hdynA) "we guided" or even deleted in some cases as in هدى+ وا <- هدوا (hdY+wA, hdwA) "they guided", and when are attached to object pronouns, it will be changed to Alif letter as in هدى+ ها <- هداها (hdY+hA, hdAhA) "guides her", etc.

The mentioned cases are just few examples to show how complex the Arabic lemmatization is, and reveal that many cases should be considered in addition to stripping words from prefixes and suffixes to get their proper lemmatization.

3. Lmmatization and IR

IR systems normally cluster words together into groups according to three main levels: root, stem, or lemma. The root level is considered by many researchers in the IR field which leads to high recall but low precision due to language complexity. For example words يكتب، مكتبة، كتاب (yktb, mktbp, ktAb) "he writes, library, book" have the same root كتب (ktb) with the basic meaning of writing. Therefore, searching for any of these words by root, yields getting the other words which may not be desirable for many users.

Other researchers show the importance of using stem level for improving retrieval precision and recall as they capture semantic similarity between inflected words. However, in Arabic, stem patterns may not capture similar words having the same semantic meaning. For example, stem patterns for broken plurals are different from their singular patterns, e.g. the stem of the plural word أقلام (AqlAm) "pens" does not match the stem of its singular form قلم (qlm) "pen". The same applies to many imperfect verbs that have different stem patterns than their perfect verbs, e.g. the verbs استطاع، يستطيع (AstTAE, ystTyE) "he could, he can" do not match because they have different stems. Indexing using lemmatization can enhance the performance of Arabic IR systems as reported in (El-Shishtawy and El-Ghannam, 2012), and in pactice, lemmatization should be very fast and accurate to be used in IR systems.

4. Related Work

A lot of work has been done in word stemming and lemmatization in different languages, for example the famous Porter stemmer for English, but for Arabic, few works have been done especially in lemmatization, and there is no open-source code and new testing data that can be used by other researchers for word lemmatization.

Xerox Arabic Morphological Analysis and Generation (Beesley, 1996) is one of the early Arabic stemmers, and it uses morphological rules to obtain stems for nouns and verbs by looking into a table of thousands of roots.

Khoja's stemmer (Khoja, 1999) and Buckwalter morphological analyzer (Buckwalter, 2002) are other root-based analyzers and stemmers which use tables of valid combinations between prefixes and suffixes, prefixes and stems, and stems and suffixes.

Recently, MADAMIRA (Pasha et al., 2014) system has been evaluated using a blind testset of 25K words for Modern Standard Arabic (MSA) selected from Penn Arabic Tree bank (PATB). They reported an accuracy of 96.2% as the percentage of words where the chosen analysis (provided by SAMA morphological analyzer (Graff et al., 2009)) has the correct lemma.

In this paper, we present an open-source Java code to extract Arabic lemmas, and a new publicly available testset for lemmatization allowing researches to evaluate using the same dataset, and reproduce results.

5. Data Description

To make the annotated data publicly available, we selected 70 news articles from Arabic WikiNews site `https://ar.wikinews.org/wiki`. These articles cover recent news from year 2013 to year 2015 in multiple genres (politics, economics, health, science and technology, sports, arts, and culture.) Articles contain 18,300 words, and they are evenly distributed among these 7 genres with 10 articles per each.

Words were white-space and punctuation separated, and some spelling errors were corrected (1.33% of the total words) to have a very clean testset. Lemmatization was done by an expert Arabic linguist where spelling corrections were marked, and lemmas were provided with full diacritization. Sample is shown in Figure 1.

As MSA is usually written without diacritics and IR systems normally remove them from search queries and also from indexed data as a basic preprocessing step, so another column for undiacritized lemma was added. This column was used to evaluate our lemmatizer and to compare with state-of-the-art systems for lemmatization (MADAMIRA), and segmentation and surface stemming (Farasa).

The raw sentences of the testset can be downloaded from the link: `http://alt.qcri.org/~hmubarak/WikiNews-26-06-2015.txt` and the annotation for lemmatization from the link: `http://alt.qcri.org/~hmubarak/WikiNews-26-06-2015-RefLemma.xlsx`

refLemmaUndiac	refLemma	spell	corrWord	orgWord
إعلام	إِعْلَام		للإعلام	للإعلام
بديل	بَدِيل		البديل	البديل
حر	حُرّ		والحر	والحر
،	،		،	،
دعا	دَعَا	1	ودعا	ودعى
حضور	حُضُور		حضور	حضور
مؤتمر	مُؤْتَمَر		المؤتمر	المؤتمر
من	مِنْ		من	من
مهتم	مُهْتَمّ		المهتمّين	المهتمّين

Figure 1: Lemmatization of WikiNews corpus

6. System Description

We were inspired by the work done by (Darwish and Mubarak, 2016) for segmenting Arabic words out of context. They achieved an accuracy of almost 99%; slightly better than state-of-the-art system for segmentation (MADAMIRA) which considers surrounding context and many linguistic features. This system shows enhancements in both Machine Translation, and Information Retrieval tasks (Abdelali et al., 2016). This work can be considered as an extension to word segmentation.

We used a fully diacritized corpus created by a commercial vendor which contains 9.7 million words with almost 200K unique surface words. About 73% of the corpus is in MSA and covers variety of genres like politics, economy, sports, society, etc. and the remaining part is mostly religious texts written in classical Arabic (CA). (Darwish et al., 2017) used this corpus to build state-of-the-art diacritizer with word error rates (WER) of 3.29% and 12.76% in diacritization of stem and grammatical case ending in order.

From this corpus, we constructed a dictionary of words and their possible diacritizations ordered by number of occurrences of each diacritized form. For example, the word وبنود (wbnwd) "and items" is found 4 times in this corpus with two full diacritization forms وَبُنُودٍ ، وَبُنُودِ (wabunudi, wabunudK) "and items, with different grammatical case endings" which appeared 3 times and once respectively. All unique undiacritized words in this corpus were analyzed using Buckwalter morphological analyzer which gives all possible word analyses, and for each analysis it provides its diacritization, segmentation, **lemma** and part-of-speech (POS) tag as shown in Figure 2.

INPUT STRING: وبنود

LOOK-UP WORD: wbnwd

SOLUTION 1: (**wabunuwd**) [**banod**_1] wa/CONJ+bunuwd/NOUN

(GLOSS): and + articles/clauses +

SOLUTION 2: (**wabinawod**) [**nawod**_1] wa/CONJ+bi/PREP+nawod/NOUN

(GLOSS): and + by/with + swaying/swinging +

Figure 2: Buckwalter analysis (diacritization forms and lemmas are highlighted)

The idea is to take the most frequent diacritized form for words that appear in this corpus, and find the morphological analysis with highest matching score between its diacritized form and the corpus diacritized word. This means that we search for the most common diacritization of words regardless of their surrounding contexts. In the above example, the first solution is preferred and hence its lemma بند (banod, bnd after diacritics removal) "item", and the other less frequent analysis is ignored.

While comparing two diacritized forms from the corpus and Buckwalter analysis, many special cases were applied to solve inconsistencies between the two diacritization schemas, for example while words are fully diacritized in the corpus, Buckwalter analysis gives diacritics without case ending (i.e. without context), and it removes short vowels in some cases, for example before long vowels, and after the definite article ال (Al) "the", etc.

It is worth mentioning that there are many cases in Buckwalter analysis where for input word, there are two

or more identical diacritizations with different lemmas, and the analyses in these cases are provided in a random order. For example the word سيارة (syArp) "car" has two morphological analyses with different lemmas; سيار (syAr) "walker", and سيارة (syArp) "car" in this order while the second lemma is the most common one. To solve this problem, all such words were reported and the top frequent words were revised to insure that their lemmas are sorted according to actual usage in a modern large corpus [3].

The lemmatization algorithm can be summarized in Figure 3, and the code can be tested and downloaded from Farasa site: `farasa.qcri.org`. It can be called from the command line as a Java package (.jar) using the following syntax:
farasa –lemma -i <*inFilename*> **-o** <*outFilename*>

Figure 4 shows system output for a sample sentence where errors are highlighted.

7. Evaluation

Lemmatization outputs of MADAMIRA and our system were compared against the undiacritized reference lemma for each word. We evaluated also surface stemming of Farasa segmenter (Darwish and Mubarak, 2016) (i.e. the remaining part after removing prefixes and suffixes) to quantify the improvement in lemmatization accuracy after applying the suggested algorithm.

For more accurate results, all differences were revised manually to accept cases that should not be counted as errors, for example in different writings for foreign named entities such as هونج كونج and هونغ كونغ (hwng kwng, hwnj kwnj) "Hong Kong".

Table 3 shows results of testing our system, MADAMIRA and Farasa segmenter as surface stemmer on the WikiNews testset (for undiacritized lemmas). Our approach gives +7% and +32% relative gains above MADAMIRA and Farasa segmenter respectively in lemmatization task.

System	Accuracy
Farasa segmenter (surface stemmer)	73.68%
MADAMIRA	96.61%
Our lemmatization System	**97.32%**

Table 3: Lemmatization accuracy using WikiNews testset

In terms of speed, our system was able to lemmatize 7.4M words on a personal laptop in 2 minutes compared to 2.5 hours for MADAMIRA which does the full morphological analysis and disambiguation, lemmatization, POS tagging, named entity recognition, and diacritization.

The code is written entirely in Java without any external dependency which makes its integration in other systems quite simple.

[3]We used text from www.Aljazeera.net which contains 100M words (archive of 10 years)

7.1. Error Analysis

Most errors in our system are due to using only the most frequent diacritization of words without considering their contexts. This cannot solve ambiguity in cases like when nouns and adjectives share the same diacritization forms, e.g. the word أكاديمية (AkAdymyp) can be either noun and its lemma is أكاديمية (AkAdymyp) "academy", or adjective and its lemma is أكاديمي (AkAdymy) "academic".

For MADAMIRA, most errors came from selecting incorrect POS tag, hence lemma, for ambiguous words (ex: the word محاضرات (mHADrAt) "lectures" was mistakenly tagged as adjective محاضر (mHADr) "lecturer" instead of the correct noun محاضرة (mHADrp) "lecture"). Another source of errors is the wrong segmentation of named entities, ex: the word البايثون (AlbAyvwn) "the+Python" which should be segmented as ال+بايثون (Al+bAyvwn) "the Python", i.e. split the definite article ال (Al) "the".

For Farasa segmenter (surface stemmer), errors came from not supporting complex cases described in Section 2..

8. Conclusion

In this paper, we list some complexities in building lemmatization for Arabic due to its rich morphology and complex writing system. We introduce a new testset for lemmatization and a very fast and accurate algorithm that performs better than state-of-the art lemmatization system; MADAMIRA. It also outperforms state-of-the-art word segmenter (surface stemmer); Farasa segmenter.

From a large fully diacritized corpus, possible diacritizations of words are extracted, and the algorithm considers only the most frequent diacritized form for words out of context. It gets the best similarity matching score between this diacritized form and the morphological analysis, provided by Buckwalter morphological analyzer, which contains word lemma. Both the testset and the code are publicly available for researchers.
We plan to study the performance when we consider sourrouning context, also to provide diacritized lemmas which can be useful for other applications. In addition, we plan to plug the lemmatizer into an IR system (Solr for example), and carry out an extrinsic evaluation to evaluate performance with lemmatization also versus other systems such as MADAMIRA and Farasa surface stemmer.

9. Bibliographical References

Abdelali, A., Darwish, K., Durrani, N., and Mubarak, H. (2016). Farasa: A fast and furious segmenter for arabic. In *HLT-NAACL Demos*, pages 11–16.

Beesley, K. (1996). Arabic finite-state morphological analysis and generation. In *In COLING-96: Proceedings of the 16th international*, pages 89–94.

Buckwalter, T. (2002). Arabic finite-state morphological analysis and generation. In *http://members.aol.com/ArabicLexicons/*.

```
Remove diacritics from input buffer
For each word in the input buffer
    Search for the word in dictionary that contains words, diacritizations, and lemmas
    If found
        Take the first lemma
    else
        Segment word into clitics (using Farasa segmentation)
        Skip prefixes from the segmented word and get the first clitic (mostly stem) after prefixes
        Handle special cases for some suffixes, e.g. taa marbouta and Hamza on yaa and waw
        Search for the stem in the dictionary
        if found
            Take the first lemma
        else
            Lemma = stem
```

Figure 3: Summary of lemmatization algorithm

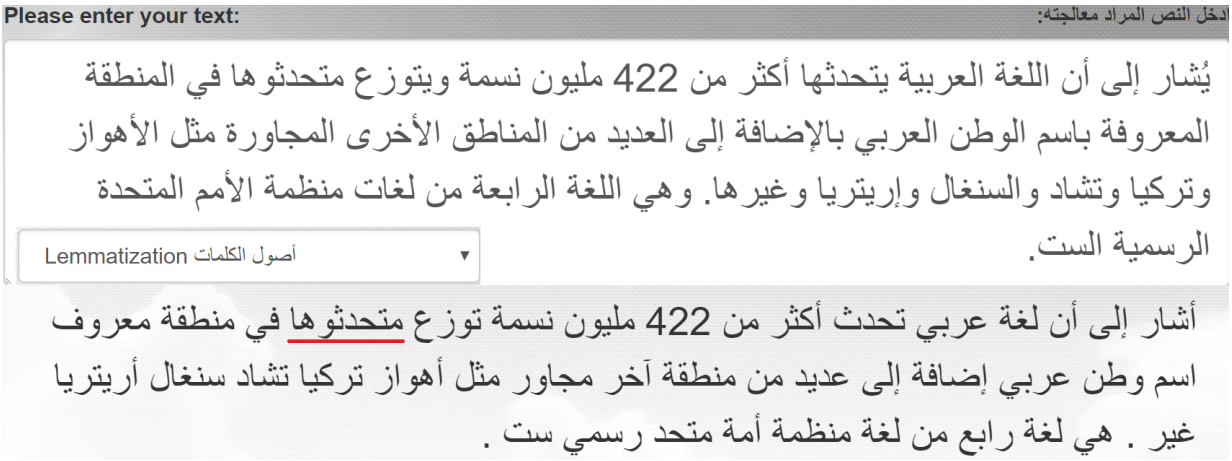

Figure 4: Lemmatization online demo (part of Farasa Arabic NLP tools)

Darwish, K. and Mubarak, H. (2016). Farasa: A new fast and accurate arabic word segmenter. In *LREC*.

Darwish, K., Mubarak, H., and Abdelali, A. (2017). Arabic diacritization: Stats, rules, and hacks. In *Proceedings of the Third Arabic Natural Language Processing Workshop*, pages 9–17.

Dichy, J. and Fargaly, A. (2003). Roots and patterns vs. stems plus grammar-lexis specifications: on what basis should a multilingual lexical database centred on arabic be built? In *Proceedings of the MTSummit, New-Orleans*.

El-Shishtawy, T. and Al-Sammak, A. (2009). Arabic keyphrase extraction using linguistic knowledge and machine learning techniques. In *Proceedings of the Second International Conference on Arabic Language Resources and Tools, The MEDAR Consortium*.

El-Shishtawy, T. and El-Ghannam, F. (2012). An accurate arabic root-based lemmatizer for information retrieval purposes. *arXiv preprint arXiv:1203.3584*.

Graff, D., Maamouri, M., Bouziri, B., Krouna, S., Kulick, S., and Buckwalter, T. (2009). Standard arabic morphological analyzer (sama) version 3.1. In *Linguistic Data Consortium LDC2009E73*.

Khoja, S. (1999). Stemming arabic text. In *Computing Department, Lancaster University*.

Pasha, A., Al-Badrashiny, M., Diab, M., El Kholy, A., Eskander, R., Habash, N., Pooleery, M., Rambow, O., and Roth, R. M. (2014). Madamira: A fast, comprehensive tool for morphological analysis and disambiguation of Arabic. *Proc. LREC*.

Plisson, J., Lavrac, N., and Mladenic, M. (2004). A rule based approach to word lemmatization. In *researchgate.net*.