Spline functions for Arabic morphological disambiguation

Mohamed Boudchiche and Azzeddine Mazroui
Department of Mathematics and Computer Science, Faculty of Sciences, Mohamed First University, Oujda, Morocco

Received 24 October 2019
Revised 5 January 2020
Accepted 21 February 2020

Abstract
We have developed in this paper a morphological disambiguation hybrid system for the Arabic language that identifies the stem, lemma and root of a given sentence words. Following an out-of-context analysis performed by the morphological analyser Alkhalil Morpho Sys, the system first identifies all the potential tags of each word of the sentence. Then, a disambiguation phase is carried out to choose for each word the right solution among those obtained during the first phase. This problem has been solved by equating the disambiguation issue with a surface optimization problem of spline functions. Tests have shown the interest of this approach and the superiority of its performances compared to those of the state of the art.

Keywords Natural Language Processing, AlKhalil Morpho Sys, Spline functions, Morphological disambiguation, Stemming
Paper type Original Article

1. Introduction

One of the main challenges facing Natural Language Processing (NLP) is the presence of ambiguity in a more or less important set of words depending on the language. A word is ambiguous if its analysis provides more than one solution. Thus, a word is morphologically ambiguous if it can accept several segmentations in (proclitic + stem + enclitic) or different morphological tags (for example, several stems or lemmas or roots). Recall that the root of a word is an abstract unit representing its origin. The lemma constitutes the minimal lexical unit having a meaning and obtained by derivation of a root according to a scheme. Finally, the stem is an inflected form of a lemma [1]. Similarly, a word is syntactically ambiguous if it can have several syntactic functions (for example, to be subject or object). Finally, a word is semantically ambiguous if it has several meanings [2].

Ambiguity is very present in the Arabic language because of its agglutinating and derivational characteristics [3]. Moreover, the absence of diacritical marks in the vast majority of Arabic texts greatly amplifies ambiguity [4]. For example, the non-diacritized

Declaration of Competing Interest: The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Publishers note: The publisher wishes to inform readers that the article “Spline functions for Arabic morphological disambiguation” was originally published by the previous publisher of *Applied Computing and Informatics* and the pagination of this article has been subsequently changed. There has been no change to the content of the article. This change was necessary for the journal to transition from the previous publisher to the new one. The publisher sincerely apologises for any inconvenience caused. To access and cite this article, please use Boudchiche, M., Mazroui, A. (2020), “Spline functions for Arabic morphological disambiguation”, *Applied Computing and Informatics*. Vol. ahead-of-print No. ahead-of-print. https://10.1016/j.aci.2020.02.002. The original publication date for this paper was 29/02/2020.

Applied Computing and Informatics
Vol. 20 No. 3/4, 2024
pp. 198-216
Emerald Publishing Limited
e-ISSN: 2210-8327
p-ISSN: 2634-1964
DOI 10.1016/j.aci.2020.02.002

word "فرمت" /*frmt*/[1] may be the verb "فَرَمَتْ" /*faramato*/ that has two meanings depending on its context: (and she threw) whose root is "ر م ي" /*r m y*/, or (she cut) whose root is "ف ر م" /*frm*/. Likewise, this word can be the verb "فُرِمَتْ" /*furimato*/ (it was cut off) whose root is "ف ر م" /*f r m*/. Habash [5] argues that on average a non-diacritized word can have 12 different morphological analyses. Similarly, in the statistical study conducted in [4] on a corpus of more than 82 million non-diacritized words, the authors showed that a word analysed out of context has on average 2.63 roots, 5.11 lemmas, 4.79 stems and 3.86 POS tags.

Morphological analysis is an essential step in the vast majority of NLP applications (text classification, machine translation, indexing, opinion mining, etc.). Indeed, these applications use canonical forms of the words to be analysed (stem, lemma, or root) to better represent them, and thereafter improve the performance of information retrieval systems [6,7].

Most of the disambiguation systems developed for the Arabic language proceed in two phases. In the first, the system performs a morphological analysis of the words taken out of their contexts. Thus, the system provides all possible morphological analyses of each word. Then, in the second phase, the system proceeds to a disambiguation phase, which consists in choosing the most appropriate solution among those proposed in the first phase. Methods used in the disambiguation phase can be probabilistic (HMM, Maximum entropy, N-grams, SVM) or neural.

In this contribution, we propose a new method of disambiguation of Arabic texts. We start with a morphological analysis out of context of the words of the sentence. For this, we use the second version of the morphological analyser Alkhalil Morpho Sys [8]. Then, we propose in the disambiguation phase a new approach based on spline functions [9]. This approach consists of equating the disambiguation issue with a surface optimization problem of spline functions.

The use of splines in the disambiguation phase has several advantages over statistical methods. Indeed, the simplicity of their expressions makes their implementation very easy. In addition, the performances of the spline disambiguation systems in terms of accuracy and speed are better than those based on HMM or SVM. Finally, the use of splines exempts us from resorting to the smoothing methods widely solicited in the training phases of the statistical approaches to circumvent the problem of absence of some words or transitions in the training corpus.

The paper is organized as follows. We give in the second section a state of art on the Arabic morphological disambiguation. Then, we briefly present in the third section the morphological analyser Alkhalil Morpho Sys, the Nemlar corpus used in the training and testing phases, and we recall the definition and some properties of the spline functions. We give in the fourth section a description of our system, and we reserve the following section for an evaluation of its performances. We end the paper with a conclusion and some perspectives.

2. Related work

We distinguish two classes of disambiguation systems: systems that are limited to the study of a single morphological tag, and those which analyse several morphological tags. The approaches developed in these disambiguation systems are either rule-based, or entirely statistical, or hybrid using both linguistic rules and statistical processing.

Given the interest of roots in many NLP applications, many researchers have been interested in the development of a root extractor (heavy stemmer). The work of [10] is one of the first works in this field. Their system is rule-based and consists of first eliminating the clitics of the word and then deducing the root. Yousef et al., [11] adopted a statistical approach to develop a heavy stemmer. They used transducers and rational kernels to model the word patterns of the Arabic language, and thereby extract a single root for each analysed word. Similarly, Boudlal et al., [12] developed a hybrid method for root extraction. The first step of this system consists of an out of context morphological analysis of the words, and the second step is a disambiguation phase based on the HMM.

The research on stems (light stemming) has also interested many researchers and several light stemmers have been previously developed. Indeed, Larkey et al., [13] developed an evolved version of light stemmers previously developed by the same authors [14]. They used a set of rules to eliminate clitics attached to words. They then proved the effectiveness of this light stemmer by testing it in the field of information retrieval. Ababneh et al., [15] have also developed a rule-based light stemmer. They have thus exploited the rules to solve some ambiguity problems.

The extraction of lemmas for the Arabic language has aroused interest only in recent years. El-shishtawy et al., [16] proposed a system based on linguistic rules and resources. The system begins by searching for the stem and the pattern of the word. Then, it uses the pattern to identify the POS tag, and exploits this information to extract the word lemma. Similarly, [17] presented a hybrid method for the extraction of lemmas. It consists in using a learning classifier that can predict the lemma pattern of a given stem, and then retrieve the lemma using this pattern and some rules.

Other works have recently focused on the analysis of several tags instead of just one. MADAMIRA is a hybrid system of morphological disambiguation widely used by researchers [18]. After out of context morphological analysis, the system uses SVM and language models in the disambiguation phase. The tags obtained following the analysis of a word by this system are its vocalized form, its lemma, its stem and its POS tag. CamelParser is an Arabic syntactic dependency analysis system aligned with contextually disambiguated morphological features [19]. Based on MADAMIRA, it improves its results and generates syntactically enriched syntactic dependencies. Similarly, Bounhas et al., [20] developed an approach combining linguistic rules and statistical classification to morphological disambiguation of non-vocalized Arabic texts. They first performed unsupervised training from unlabelled vocalized Arabic corpora. Then, to deal with imperfect data, they compared a possibilistic approach to a data transformation-based approach. Experiments have shown that on classical texts, the possibilistic approach applied to 14 morphological features gives better results than the one based on transformation. The system Bel-Arabi developed in [21] is a grammar analyser based on basic morphology analysis and some Arabic grammar rules. This system provides the stem, the POS tag and the base phrase chunking. Farasa is a morphosyntactic disambiguation system developed by QCRI Arabic Language Technologies [22]. It is an open-source tool and consists of a segmentation module, a POS tagger, an Arabic text Diacritizer and a Dependency Parser. The segmentation module is based on SVM-ranking model [23], while the POS tagger and Dependency Parser are based on the randomized greedy algorithm [24].

3. Tools and corpora

3.1 Alkhalil Morpho Sys 2

AlKhalil Morpho Sys 2 is an open source morphosyntactic analyser developed by the Computer Research Laboratory of Mohammed First University, Morocco [8]. It analyses both non vocalized Arabic words and partially or completely vocalized words. The analysis is done out of context and the results for a given word are:

- the possible vocalized forms of the word,
- for each possible vocalized form, the system provides the segmentation of the word in proclitic + stem + enclitic, its POS tag, its syntactic state (case for nouns and mood for verbs), its lemma and its stem accompanied by their patterns.

We used this analyser in the first phase of our system and we are just interested in stem, lemma and root tags.

3.2 Training and testing corpora

The Nemlar project (Network for Euro-Mediterranean Language Resources) started in 2003 in the framework of the the MED-Unco program supported by the European Union [25], which brought together 14 partners from various countries. It was aimed at developing the resources of the Arabic language. Nemlar corpus is a set of Arabic-language texts originally annotated by the Egyptian company RDI on behalf of the Nemlar consortium that holds the rights. It was collected from 13 different domains spread across 489 files that contain about 500,000 words.

This corpus has been recently corrected and enriched by other tags. Its latest version is open and can be downloaded at the following address: http://oujda-nlp-team.net/en/corpora/nemlar-corpus-en/. The tags provided for a given word are:

- its vocalized form,
- its stem,
- the clitics attached to the stem,
- its lemma,
- its grammatical category,
- its scheme.

To build our systems and evaluate them, we segmented this corpus into ten parts of the same size. The segmentation was performed randomly at the sentence level. Nine parts (about 90% of the corpus) will constitute the training set E_A, which will be used in the training phases of our systems. The test set E_T, consisting of the remaining 10% of the Nemlar corpus and containing around 50,000 words, will be used in the testing phases.

3.3 Spline functions

A spline function ϕ is a piecewise polynomial function on an [a, b] interval [9]. Thus, ϕ is associated with a subdivision $(x_i)_{(1 \le i \le m)}$ of [a, b] so that the restriction of ϕ to each $[x_i, x_{i+1}]$ interval is a Polynomial ($(x_i)_{(1 \le i \le m)}$ are also called the knots of the spline).

The spline ϕ is of class C^r and degree n if it is of class C^r over the whole [a, b] interval, and its restriction to each $[x_i, x_{i+1}]$ interval, $1 \le i < m$ is a Polynomial of degree n.

In general, spline functions are used to approximate a function or interpolate data. In the following, we will recall the classical theorem of interpolation theory.

Theorem 1 Given two dots c and d of the [a, b] interval and a set of data $(f_i)_{(1 \le i \le k_1)}$ and $(g_i)_{(1 \le i \le k_2)}$, there exists a unique polynomial P of degree $(1 + k_1 + k_2)$ and verifying the following Hermite interpolation conditions:

$$P(c) = f_0;\ P^{(i)}(c) = f_i \text{ for } 1 \le i \le k_1 \tag{1}$$

$$P(c) = f_0;\ P^{(i)}(c) = g_j \text{ for } 1 \le j \le k_2 \tag{2}$$

where $h^{(i)}$ denotes the derivative of order i of the function h. $(f_i)_{(1 \le i \le k_1)}$ and $(g_i)_{(1 \le i \le k_2)}$ are called Hermite interpolation data.

4. Description of the proposed approach

In order to improve their performances, many Arabic NLP applications do not seek to extract information directly from text words. Indeed, these applications begin by replacing the words by one of their canonical forms (stem or lemma or root), and then proceed to the analysis of these latter. Thus, we are interested in this work by the extraction process of these three tags: stem, lemma and root.

The proposed approach is composed of two modules. The first one is reserved for an out of context morphological analysis of the sentence words. Thus, the system provides for each analysed word the list of its possible solutions. Then, the system uses in the second module the spline functions to eliminate the ambiguity by choosing for each word the right stem among these solutions (see Figure 1)

4.1 Morphological analysis

After a pre-processing step that consists of segmenting the text into sentences and then the sentences into words, the system performs for each sentence an out of context morphological analysis of its words. The morphological analyser Alkhalil Morpho Sys 2 was used for this task. Thus, we obtain the different possible analyses of each word (see example in Figure 2).

4.2 Disambiguation phase by spline functions

The disambiguation phase will be carried out tag by tag according to the same process. We will present in the following our stem disambiguation approach; the other tags (lemma and root) will be treated in the same way.

Given a sentence S consisting of the words $w_1, w_2, \ldots, w_k$, the morphological analysis of the first phase gives us the possible stems of each word of the sentence (see Figure 3).

The disambiguation phase consists in identifying the path of the correct stems $(s_1^*, \ldots, s_k^*)$ of the words $(w_1, \ldots, w_k)$ among the $(n_1 \times \ldots \times n_k)$ possible paths $\left(s_{j_1}^1, \ldots, s_{j_k}^k\right)$, where $s_{j_1}^1 \in S_i = \{s_1^i, \ldots, s_{n_i}^i\}$ the set of the potential stems of the word w_i.

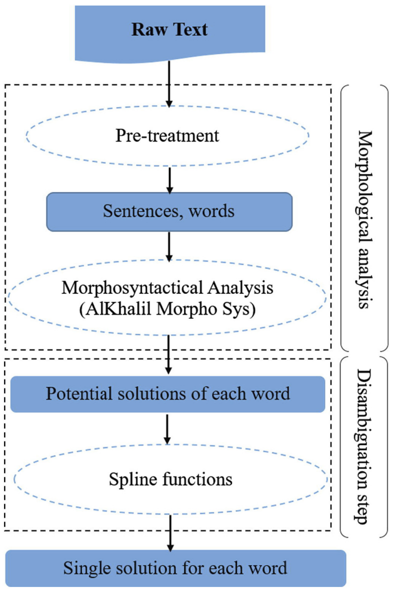

Raw Text
Pre-treatment
Sentences, words
Morphosyntactical Analysis (AlKhalil Morpho Sys)
Morphological analysis
Potential solutions of each word
Spline functions
Disambiguation step
Single solution for each word

Figure 1.
System architecture.

To identify this optimal path, we will associate to each potential path $\left(s^1_{j_1}, \ \ldots, \ s^k_{j_k}\right)$ a spline $\psi_{s^1_{j_1},\ldots,s^k_{j_k}}$ on the [1, k] interval so that the spline $\psi_{s^*_1,\ldots,s^*_k}$ associated with the optimal path $(s^*_1, \ldots, s^*_k)$ has a maximum area, i.e.:

$$\int_1^k \psi_{s^*_1,\ldots,s^*_k}(x)dx = \max_{\substack{1\leq j_l \leq n_l \\ 1 \leq l < k}} \int_1^k \psi_{s^1_{j_1},\ldots,s^k_{j_k}}(x)dx$$

Given that all paths are composed of the same number of stems, and since it is the position of the stem in the path that is important, we have assimilated the stems to their positions in the path to construct the associated spline. Thus, the knots of the spline $\psi_{s^1_{j_1},\ldots,s^k_{j_k}}$ associated with a given path $\left(s^1_{j_1}, \ \ldots, \ s^k_{j_k}\right)$ are the integers between 1 and k.

```
<?xml version="1.0" encoding="UTF-8" ?>
<analysis>
    <result word='فرمت' nbresult='28'>
        <morph_feature_set diac="فَرَمَتْ"
            stem="فَرَمَتْ" lemma="رَمَى" root="رمي"
            pat_lemma="فَعَلَ" pat_stem="فَعَلَتْ"
            pos="فعل|ماض|-|معلوم|ثلاثي|مجرد|غائب|هي|لازم ومتعد" cas="-"
            proc="فَ|حرف العطف أو الاستئناف" enc="ت|تاء التأنيث الساكنة"
        />
        <morph_feature_set diac="فُرِمَتْ"
            stem="فُرِمَتْ" lemma="فَرَمَ" root="فرم"
            pat_lemma="فَعَلَ" pat_stem="فُعِلَتْ"
            pos="فعل|ماض|-|مجهول|ثلاثي|مجرد|غائب|هي|متعد" cas="-"
            proc="#" enc="ت|تاء التأنيث الساكنة"
        />
```

Figure 2. Some analysis solutions of the word “فرمت”.

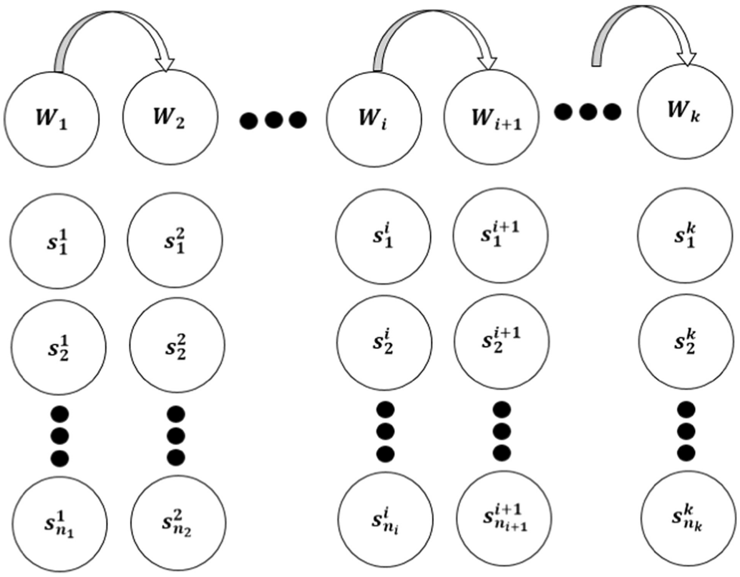


Figure 3. Possible stems of the sentence words.

In order to evaluate the impact of using the context in the disambiguation phase, we will experiment with three families of splines. The first one concerns linear splines built solely from information on the stems of words and without the use of information on their contexts. The quadratic splines of the second family exploit, in addition to information used in the construction of linear splines, the left context (in the order of the Arabic script) of the analysed words. Finally, the construction of the cubic splines of the third family is based on the exploitation of the right context of the analysed words in addition to the information above.

4.2.1 Disambiguation by linear splines. The linear spline associated with a potential path $(s_1, \ldots, s_k)(s_i \in S_i)$ is obtained by choosing as Hermite interpolation data at each knot i, $1 \le i \le k$, the weight in the Arabic language of the stem s_i. Thus, the spline $\psi^L_{(s_1,\ldots,s_k)}$ associated with this path satisfies the following Hermite interpolation conditions:

$$\psi^L_{(s_1,\ldots,s_k)} = p_i \quad \text{for } 1 \le i \le k \tag{3}$$

where p_i is the weight in the Arabic language of the stem s_i.

The following result is a consequence of Theorem 1 above.

Theorem 2 Given a sequence of values $(p_i)_{(1 \le i \le k)}$, there is a single linear continuous spline $\psi^L_{(s_1,\ldots,s_k)}$ (of degree 1 and class C^0 in [1, k]), checking in each [i, i+1] interval, $1 \le i < k$, the following interpolation conditions:

$$\psi^L_{(s_1,\ldots,s_k)}(i) = p_i \text{ and } \psi^L_{(s_1,\ldots,s_k)}(i+1) = p_{i+1} \tag{4}$$

The expression of $\psi^L_{(s_1,\ldots,s_k)}$ in the [i, i + 1] interval is given by:

$$\psi^L_{(s_1,\ldots,s_k)}(x) = p_i + (p_{i+1} - p_i)(x - i) \tag{5}$$

and

$$\int_i^{i+1} \psi^L_{(s_1,\ldots,s_k)}(x)dx = \frac{1}{2}(p_i + p_{i+1}) \tag{6}$$

It is easy to verify that if the Hermite data $(p_i)_{(1 \le i \le k)}$ are positive, then the spline $\psi^L_{(s_1,\ldots,s_k)}$ is positive.

We note that this spline is built solely from information on the stems and does not exploit the context of the associated words.

4.2.2 Disambiguation by quadratic splines. The quadratic spline relative to a potential path $(s_1, \ldots, s_k)$ is obtained by choosing as the value at the knot $i \in \{1, \ldots, k\}$ the weight in the Arabic language of the stem s_i and as right derivative the weight in the Arabic language of the transition from the stem s_i to the stem s_{i+1}. Thus, by applying Theorem 1 above, we have the following result.

Theorem 3 Given two series of values $(p_i)_{(1 \le i \le k)}$ and $(t_i)_{(1 \le i < k)}$, there exists a single continuous quadratic spline $\psi^Q_{(s_1,\ldots,s_k)}$ (of degree 2 and of class C^0 in [1, k]) satisfying in each [i, i+1] interval, $1 \le i < k$, the following interpolation conditions:

$$\begin{aligned} &\psi^Q_{(s_1,\ldots,s_k)}(i) = p_i \\ &\psi^Q_{(s_1,\ldots,s_k)}(i+1) = p_{i+1} \\ &D_r\psi^Q_{(s_1,\ldots s_k)}(i) = t_i \end{aligned} \tag{7}$$

where $D_r g(x)$ is the right derivative of the function g at the point$x$.

The expression of the spline $\psi^{Q}_{(s_1,\ldots,s_k)}$ in the [i, i + 1] interval is given by:

$$\psi^{Q}_{(s_1,\ldots,s_k)}(x) = p_i + t_i(x-i) - (p_i - p_{i+1} + t_i)(x-i)^2 \tag{8}$$

As for the linear spline, it is easy to verify that if the Hermite data $(p_i)_{(1\le i\le k)}$ and $(t_i)_{(1\le i<k)}$ are positive, then the spline is positive.

Furthermore,

$$\int_i^{i+1} \psi^{Q}_{(s_1,\ldots,s_k)}(x)dx = \frac{2}{3}p_i + \frac{1}{3}p_{i+1} + \frac{1}{6}t_i \tag{9}$$

4.2.3 Disambiguation by cubic splines. In addition to the values at knots i and $i+1$ and the right derivative at knot i necessary for the construction of the quadratic spline, the cubic spline also interpolates the left derivative at knot $i+1$.

Theorem 4 Given a potential path $(s_1, \ldots, s_k)$ and three value sequences $(p_i)_{(1\le i\le k)}$,$(t_i)_{(1\le i<k)}$ and $(T_i)_{(1<i\le k)}$, there is a single continuous cubic spline $\psi^{C}_{(s_1,\ldots,s_k)}$ (of degree 3 and of class C^0) satisfying in each [i, i+1] interval, $1\le i<k$, the following interpolation conditions:

$$\begin{aligned} \psi^{C}_{(s_1,\ldots,s_k)}(i) &= p_i \\ \psi^{C}_{(s_1,\ldots,s_k)}(i+1) &= p_{i+1} \\ D_r\psi^{C}_{(s_1,\ldots,s_k)}(i) &= t_i \\ D_l\psi^{C}_{(s_1,\ldots,s_k)}(i+1) &= T_{i+1} \end{aligned} \tag{10}$$

where $D_lg(x)$ is the left derivative of the function g at the point x.

The expression of the spline $\psi^{C}_{(s_1,\ldots,s_k)}$ in the [i, i + 1] interval is given by:

$$\begin{aligned} \psi^{C}_{(s_1,\ldots,s_k)}(x) = p_i + t_i(x-i) &- (3p_i - 3p_{i+1} + 2t_i + T_{i+1})(x-i)^2 \\ &+ (2p_i - 2p_{i+1} + t_i + T_{i+1})(x-i)^3 \end{aligned} \tag{11}$$

and

$$\int_i^{i+1} \psi^{C}_{(s_1,\ldots,s_k)}(x)dx = \frac{p_i + p_{i+1}}{2} + \frac{t_i - T_{i+1}}{12} \tag{12}$$

Unlike the linear and quadratic splines, the cubic spline does not maintain the positivity of the data. Indeed, even if the Hermite data $(p_i)_{(1\le i\le k)}$,$(t_i)_{(1\le i<k)}$ and $(T_i)_{(1<i\le k)}$ are all positive, the spline $\psi^{C}_{(s_1,\ldots,s_k)}$ is not necessarily positive. For example, if we take on the [1,2] interval the following data $p_1 = 0.4$, $p_2 = 0.01$, $t_1 = 0.6$ and $T_2 = 0.8$, then the associated cubic spline is negative in the [1.75, 1.95] interval.

4.3 Estimation of model parameters

To estimate the different Hermite data defining the models (i.e. the three sequences $(p_i)_{(1\le i\le k)}$,$(t_i)_{(1\le i<k)}$ and $(T_i)_{(1<i\le k)}$, we use a labeled training corpus C, and we propose several choices based on the maximum likelihood [26].

4.3.1 Estimation of stem weights. The weight p_i of a stem s_i associated with a word w_i can be estimated by one of the following two formulas:

$$p_i = \frac{\mathrm{Occ}(\mathrm{w}_i, s_i)}{\mathrm{Occ}(\mathrm{w}_i)} \tag{P1}$$

or

$$p_i = \frac{\mathrm{Occ}(s_i)}{\sum_{l=1}^{n_i}\mathrm{Occ}\left(s_l^i\right)} \tag{P2}$$

where

- $\mathrm{Occ}(\mathrm{w}_i, s_i)$ = number of occurrences in the training corpus C of the word w_i associated with the stem s_i,
- $\mathrm{Occ}(\mathrm{w}_i)$ = number of occurrences in the training corpus C of the word w_i,
- $\mathrm{Occ}(s_i)$ = number of occurrences in the training corpus C of the stem s_i,
- $\{s_l^i, 1 \le l \le n_i\}$ is the set of the potential stems of the word w_i proposed by the morphological analyser.

(P1) is none other than the estimate of the probability that the word w_i is associated with the stem s_i in the corpus C. However, (P2) represents the frequency of the stem s_i in the set of potential stems $\{s_1^i, \ldots, s_{n_i}^i\}$ of the word w_i.

4.3.2 Estimation of right derivatives. The right derivative at knot i is equal to the weight t_i of the transition between the stems s_i and s_{i+1} associated respectively with the words w_i and w_{i+1}. To estimate it, we propose one of the following three formulas:

$$t_i = \frac{\mathrm{Occ}(s_i, s_{i+1})}{\mathrm{Occ}(s_i)} \tag{Tr1}$$

$$t_i = \frac{\mathrm{Occ}(\mathrm{w}_i, s_i)}{\sum_{r=1}^{n_i}\sum_{u=1}^{n_{i+1}}\mathrm{Occ}\left(s_r^i, s_u^{i+1}\right)} \tag{Tr2}$$

$$t_i = \frac{\mathrm{Occ}((\mathrm{w}_i, s_i), (\mathrm{w}_{i+1}, s_{i+1}))}{\mathrm{Occ}(\mathrm{w}_i, s_i)} \tag{Tr3}$$

where

$\mathrm{Occ}(s_i, s_{i+1})$ = number of occurrences in the training corpus C of the stem s_i followed by the stem s_{i+1},

$\mathrm{Occ}((w_i, s_i), (w_{i+1}, s_{i+1}))$ = number of occurrences in the training corpus C of the word w_i associated with the stem s_i followed by the word w_{i+1} associated with the stem s_{i+1}.

Remark: the formula (Tr1) calculates an estimate of the weight of the transition t_i just from information on the stems s_i and s_{i+1}, and without taking into account the associated words w_i and w_{i+1}. While the estimate of the formula (Tr2) partially takes into account the words w_i and w_{i+1} since the normalization (the denominator value) is computed from information of the potential stems of these two words. Finally, the last formula (Tr3) estimates the transition between two stems of two consecutive words by limiting themselves just to the frequencies of appearance in the training corpus C of these words accompanied by their stems.

4.3.3 Estimation of left derivatives. To estimate the left derivative T_{i+1} at knot $(i+1)$, we considered a convex combination between the weight of the transition t_i from the stem s_i to the stem s_{i+1} and that of the transition t_{i+1} from the stem s_{i+1} to the stem s_{i+2}. So,

$$T_{i+1} = \alpha \mathrm{t}_i + (1-\alpha)\mathrm{t}_{i+1} \tag{Cu}$$

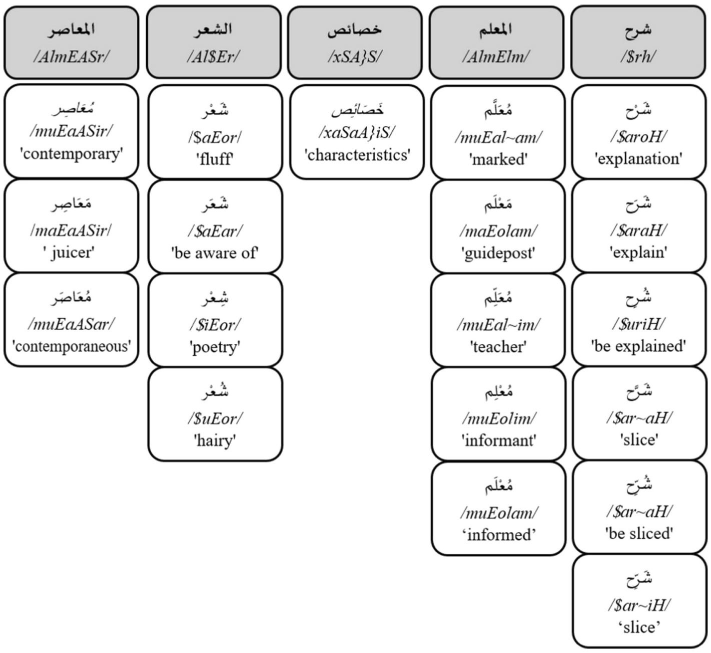
with α is a parameter to choose between 0 and 1.

If we choose $\alpha = 0$, then T_{i+1} in each knot $(i+1)$ is equal to the right derivative t_{i+1}, so the spline $\psi^{C}_{(s_1,\ldots,s_k)}$ will be of class C^1 on the [1, k] interval.

On the other hand, if $\alpha = 1$ then the left derivative T_{i+1} of the spline $\psi^{C}_{(s_1,\ldots,s_k)}$ at knot $(i+1)$ is equal to the right derivative t_i at knot i. In this case $\int_i^{i+1} \psi^{C}_{(s_1,\ldots,s_k)}(x)dx = \int_i^{i+1} \psi^{L}_{(s_1,\ldots,s_k)}(x)dx$, and this implies that the optimal path corresponding to the cubic spline is identical to that corresponding to the linear spline.

For more readability, we consider the following example "شرح المعلم خصائص الشعر المعاصر" */$rH AlmElm xSA}S Al$Er AlmEASr/* (The teacher explained the characteristics of contemporary poetry). We present in Figure 4 the potential stems of each word obtained by the morphological analysis of the first step.

We then used the relation (8) to calculate the expressions of the two quadratic splines associated with the two following possible paths "شَرْح – مُعَلم – خَصَائِص – شَعْر - مَعَاصِر" and "شَرَح – مُعَلم – خَصَائِص – شِعْر - مُعَاصِر". We have used the equations (P1) and (Tr2) to estimate the Hermite data. The surfaces of these two splines are given by the relation (9) and equal to 1.92 and 3.41 respectively. We present in Figure 5 the graphs of these two splines.



المعاصر /AlmEASr/	الشعر /Al$Er/	خصائص /xSA}S/	المعلم /AlmElm/	شرح /$rh/
مُعَاصِر /muEaASir/ 'contemporary'	شَعْر /$aEor/ 'fluff'	خَصَائِص /xaSaA}iS/ 'characteristics'	مُعَلَّم /muEal~am/ 'marked'	شَرْح /$aroH/ 'explanation'
مَعَاصِر /maEaASir/ ' juicer'	شَعَر /$aEar/ 'be aware of'		مَعْلَم /maEolam/ 'guidepost'	شَرَح /$araH/ 'explain'
مُعَاصَر /muEaASar/ 'contemporaneous'	شِعْر /$iEor/ 'poetry'		مُعَلِّم /muEal~im/ 'teacher'	شُرِح /$uriH/ 'be explained'
	شُعْر /$uEor/ 'hairy'		مُعْلِم /muEolim/ 'informant'	شَرَّح /$ar~aH/ 'slice'
			مُعْلَم /muEolam/ 'informed'	شُرِّح /$ar~aH/ 'be sliced'
				شَرِّح /$ar~iH/ 'slice'

Figure 4. Results of morphological analysis of sentence words.

4.4 Viterbi algorithm

We recall that the disambiguation phase consists of looking for the optimal path $(s_1^*, \ldots, s_k^*)$ of stems associated with the words $(w_1, \ldots, w_k)$ among the $(n_1 \times \ldots \times n_k)$ possible paths $(s_{j_1}^1, \ldots, s_{j_k}^k)$, where $s_{j_i}^i \in S_i = \{s_1^i, \ldots, s_{n_i}^i\}$ the set of the potential stems of the word w_i (see Figure 3).

This path checks the following optimization equation:

$$\int_1^k \psi_{s_1^*,\ldots,s_k^*}(x)dx = \max_{\substack{s_{j_i}^i \in S_i \\ 1 \le i < k}} \left(\int_1^k \psi_{s_{j_1}^1,\ldots,s_{j_k}^k}(x)dx \right) \tag{13}$$

where $\psi_{s_{j_1}^1,\ldots,s_{j_k}^k}$ is the spline associated with the path of stems $(s_{j_1}^1, \ldots, s_{j_k}^k)$.

To optimize the search time of this path, we have developed an algorithm inspired by that of Viterbi [27].

In what follows, we will work with the following notations:

Given two potential stems s_u^i and s_v^{i+1} $(1 \le i < k,\ 1 \le u \le n_i$ et $1 \le v \le n_{i+1})$ associated with the two words w_i and w_{i+1}, we denote by $\phi_{u,v,i}$ the polynomial that interpolates the data of the stems s_u^i and s_v^{i+1} ($\phi_{u,v,i}$ corresponds to one of the polynomials given by the equations (5) or (8) or (11)) and $I_{u,v,i} = \int_i^{i+1} \phi_{u,v,i}(x)dx$.

It is obvious that $\phi_{j_i,j_{i+1},i}$ is none other than the restriction of $\psi_{s_{j_1}^1,\ldots,s_{j_k}^k}$ at the [i, i + 1] interval and

$$\int_1^k \psi_{s_{j_1}^1,\ldots,s_{j_k}^k}(x)dx = \sum_{i=1}^{k-1} I_{j_i,j_{i+1},i} \tag{14}$$

For a partial path of potential stems $(s_{j_1}^1, \ldots, s_{j_{i-1}}^{i-1}, s_u^i)$ leading to the stem s_u^i of the word w_i, $1 \le i \le k$ and $1 \le u \le n_i$, we also denote by $\psi_{s_{j_1}^1,\ldots,s_{j_{i-1}}^{i-1},s_u^i}$ the spline associated with this path.

Let $\Lambda(i, s_u^i)$ be the maximum surface on all the splines associated with the partial paths leading to the stem s_u^i:

$$\Lambda(i, s_u^i) = \max_{\substack{1 \le j_l \le n_l \\ 1 \le l \le i-1}} \int_1^i \psi_{s_{j_1}^1,\ldots,s_{j_{i-1}}^{i-1},s_u^i}(x)dx \tag{15}$$

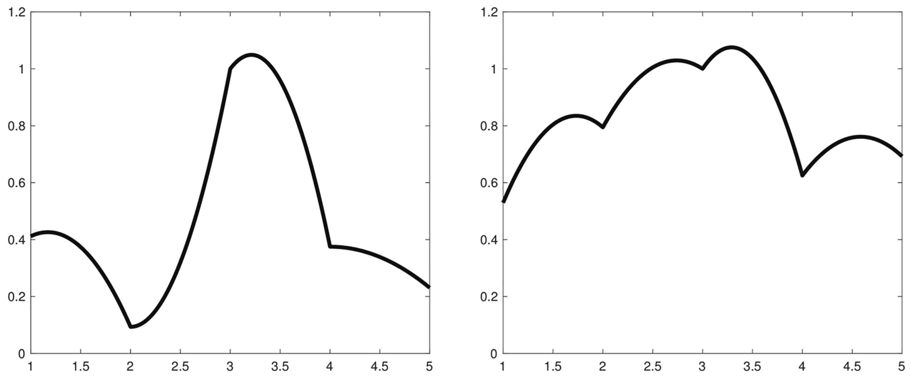


Figure 5.
On the left, the spline graph associated with the stem path (خَصَائِص – شِعْرِ - مُعَاصِر – شَرَح – مُعَلم), and on the right, that of the spline associated with the stem path (خَصَائِص – شِعْرِ - مُعَاصِر – شَرَح – مُعَلم).

It's easy to verify that

$$\Lambda\left(i, s_u^i\right) = \max_{1 \leq j_{i-1} \leq n_{i-1}} \left(\Lambda\left(i-1, s_{j_{i-1}}^{i-1}\right) + I_{j_{i-1},u,i-1}\right) \tag{16}$$

This formula will allow us to calculate the function Λ by induction.

Similarly, if we denote by $\gamma(i, s_u^i)$ the index of the stem associated with the word w_{i-1} in the optimal path leading to the stem s_u^i:

$$\gamma\left(i, s_u^i\right) = \operatorname*{argmax}_{1 \leq j_{i-1} \leq n_{i-1}} \left(\Lambda\left(i-1, s_{j_{i-1}}^{i-1}\right) + I_{j_{i-1},u,i-1}\right) \tag{17}$$

and by $\Gamma(i, s_u^i)$ the optimal path leading to the stem s_u^i, (i.e. the one with the maximum associated spline surface):

$$\Gamma\left(i, s_u^i\right) = \operatorname*{argmax}_{\substack{1 \leq j_l \leq n_l \\ 1 \leq l < i-1}} \int_1^i \psi_{j_1,\ldots,j_{i-1},u}(x)dx \tag{18}$$

then $\Gamma(i, s_u^i) \in \{1, \ldots, n_1\} \times \ldots \times \{1, \ldots, n_{i-1}\}$ and satisfies the following recurrence relation:

$$\Gamma\left(i, s_u^i\right) = \left(\Gamma\left(i-1, s_{\gamma\left(i,s_u^i\right)}^{i-1}\right), \gamma\left(i, s_u^i\right)\right) \tag{19}$$

The relations (16), (17) and (19) will allow us to identify the optimal path according to the following algorithm:

Step 1 (initialization):

Calculate for each $1 \leq u \leq n_2$:

$$\Lambda\left(2, s_u^2\right) = \max_{1 \leq j_1 \leq n_1} \left(I_{j_1,u,1}\right)$$

$$\gamma\left(2, s_u^2\right) = \operatorname*{argmax}_{1 \leq j_1 \leq n_1} \left(I_{j_1,u,1}\right)$$

and put

$$\Gamma\left(2, s_u^2\right) = \gamma\left(2, s_u^2\right)$$

Step 2 (induction):

For each $3 \leq i \leq k$ and $1 \leq u \leq n_i$, calculate $\Lambda(i, s_u^i)$, $\gamma(i, s_u^i)$ and $\Gamma(i, s_u^i)$ using the following induction relations:

$$\Lambda\left(i, s_u^i\right) = \max_{1 \leq j_{i-1} \leq n_{i-1}} \left(\Lambda\left(i-1, s_{j_{i-1}}^{i-1}\right) + I_{j_{i-1},u,i-1}\right)$$

$$\gamma\left(i, s_u^i\right) = \operatorname*{argmax}_{1 \leq j_{i-1} \leq n_{i-1}} \left(\Lambda\left(i-1, s_{j_{i-1}}^{i-1}\right) + I_{j_{i-1},u,i-1}\right)$$

$$\Gamma\left(i, s_u^i\right) = \left(\Gamma\left(i-1, s_{\gamma\left(i,s_u^i\right)}^{i-1}\right), \gamma\left(i, s_u^i\right)\right)$$

Step 3 (final state):

$$u_k^* = \underset{1 \le u \le n_k}{\operatorname{argmax}} \left(\Gamma\left(k, s_u^k\right) \right)$$

Step 4 (Optimal path):

$$\left(u_1^*, u_2^*, \ldots, u_k^*\right) = \left(\Gamma\left(k, s_{u_k^*}^k\right), u_k^*\right)$$

and

$$\left(s_1^*, s_2^*, \ldots, s_k^*\right) = \left(u_{u_1^*}^1, u_{u_2^*}^2, \ldots, u_{u_k^*}^k\right)$$

5. Results and discussion

We will first evaluate the impact of spline degree and Hermite data choices on the accuracy of the disambiguation system. This will allow us to determine how much the performances of the system are improved by the use of the context (disambiguation by the quadratic and cubic splines). Then we will make a comparison between the disambiguation system using splines and those based on HMM and SVM. This comparison is performed under identical training and testing conditions for the three systems. We end by giving the accuracy of the disambiguation systems relating to the three tags stem, lemma and root.

For all these evaluations, we used the Nemlar corpus introduced in section 3.2. We have first extracted in a random way at the level of the sentences about 90% of the corpus, noted E_A, which we used in the training phases of all the experiments realized in the two first subsections of this section. The training phase consists of estimating the Hermite data needed for calculating the splines used in the disambiguation phase. These estimates are based on equations (P1), (P2), (Tr1), (Tr2), (Tr3) and (Cu).

The set E_T, consisting of the remaining 10% of the Nemlar corpus and containing approximately 50,000 words, was used in the test phases of our systems.

The used measure of performance is the accuracy, which corresponds to the percentage of the words of the set E_T correctly labeled. It is defined by:

$$\text{accuracy} = \frac{\text{number of words of } E_T \text{ correctly labeled}}{\text{size of } E_T}$$

5.1 Impact of the spline choice

As presented in section 4.2, we can use three types of splines (linear, quadratic or cubic). Similarly, we have several choices for estimating spline parameters (section 4.3).

In the following tables, we present the results for the different possible choices.

5.1.1 Evaluation of disambiguation by linear splines. For linear splines, we have two choices for estimating the weights of the stems expressed above by the formulas (P1) and (P2).

By testing these choices on the test set E_T, we obtained the results presented in Table 1.

We find that the accuracy relative to the estimation choice of the stem weights (P1) is better than that obtained with the choice (P2). This can be explained in part by the nature of (P1), which estimates the probability that a stem is associated with a word, while (P2) estimates the weight of a stem in the set of potential stems of a given word.

5.1.2 Evaluation of disambiguation by quadratic splines. In addition to the two estimation choices of the stem weights, we have for the right derivatives of the stems three estimation choices given above by the formulas (Tr1), (Tr2) and (Tr3). The evaluation results of these different choices on the test set E_T are presented in Table 2.

Let us first note that the quadratic splines with the choice (P1) for estimating the stem weights are more performing than linear splines. Moreover, the results obtained with the choice (P1) are, as for linear splines, better than those relating to the choice (P2). Similarly, the (Tr2) estimate of the right derivatives provides the best performances. The relatively weak results obtained with the choices (Tr1) and (Tr3) can be explained by the demanding nature of the estimate (Tr3) that counts only the transitions of the words accompanied by their respective stems, and of the fact that the estimate (Tr1) is based solely on information on stems without taking into account their associated words.

5.1.3 Evaluation of disambiguation by cubic splines. To build cubic splines, we will need, in addition to the estimation of the weights and the right derivatives of the stems, to estimate their left derivatives from the above formula (Cu). Since left derivative estimates depend on a parameter $\alpha \in [0, 1]$, we performed tests on the test set E_T for a few values of α equal to the five nodes of a uniform subdivision of the interval [0,1]. We present in Table 3 the evaluation results of these tests.

These results confirm the conclusions obtained with linear and quadratic splines. Indeed, for every choice of α, the system accuracy corresponding to (P1) stem weight estimate is better than that corresponding to the (P2) estimate. Moreover, the (Tr2) estimate of the right derivatives provides the best performances. Finally, it is clear that the system performance decreases with the value of α. This implies that the more the estimate (Cu) of the left derivatives favours the right context (i.e., small α and so more weight at t_{i+1} than at t_i in (Cu) formula), the better the results. Thus, the best results are obtained for the single cubic spline of class C^1 corresponding to the choice $\alpha = 0$.

Estimate of stem weights	Accuracy
(P1)	92.39%
(P2)	91.67%

Table 1. Accuracy of the disambiguation by linear splines on the test set E_T.

	Estimate of the right derivatives		
Estimate of stem weights	(Tr1)	(Tr2)	(Tr3)
(P1)	94.06%	94.15%	93.13%
(P2)	90.18%	91.63%	89.93%

Table 2. Accuracy of disambiguation by quadratic splines on the test set E_T.

		Estimate of the right derivatives		
	Estimate of stem weights	(Tr1)	(Tr2)	(Tr3)
$\alpha = 0$	(P1)	94.005%	94.04%	92.27%
	(P2)	91.07%	92.16%	91.05%
$\alpha = 0.25$	(P1)	93.97%	94.01%	92.16%
	(P2)	90.78%	92.13%	90.75%
$\alpha = 0.5$	(P1)	93.83%	93.99%	92.11%
	(P2)	90.72%	92.10%	90.71%
$\alpha = 0.75$	(P1)	93.81%	93.93%	92.06%
	(P2)	90.70%	92.07%	90.68%

Table 3. Accuracy of the disambiguation by cubic splines for an estimation of left derivatives corresponding to different choices of α.

By comparing the results obtained by each spline according to the different choices of estimation of its parameters, we conclude that the use of the quadratic splines with (P1) stem weight estimate and (Tr2) estimate of the right derivatives provides the best performances.

5.2 Comparison of the spline-based model with other models

To evaluate the impact of spline use in the disambiguation phase, we will compare the performances of the spline-based model with two other models successively using HMM and SVM in the disambiguation phase.

In this part, we will limit ourselves to the spline-based model that has performed best, namely the model based on the quadratic spline built from the (P1) stem weight estimate and the (Tr2) right derivative estimate. We kept the morphological phase for the three models, and only modified the disambiguation phase by first using the HMM, then the SVM (for more details on these models see for example [28,29]).

The training phase of each of these three models (Spline, HMM and SVM) was carried out from the same corpus E_A, and they were tested on the same test corpus E_T.

We calculated both the accuracy of each model and the speed, which is equal to the number of analysed words per second. The obtained results are shown in Table 4.

It is clear that the model based on the quadratic splines realizes the best performances. Indeed, the accuracy of this model exceeds 94% whereas those of the models based on the HMM and the SVM reach only 92.35% and 88.43% respectively. In addition, this model is the fastest given that it analyses on average 290 words per second against only 254 for the HMM-based model and 210 for the one based on SVM. Finally, the smoothing methods used in the training phases of statistical approaches to deal with the problems of the absence of certain words or transitions in the training corpus are not essential for spline-based methods. Indeed, in HMM-based methods, the search for the solution consists of identifying the path with the highest probability. And since the probability of a path is expressed as a product of transition and emission probabilities, the absence of one of these transitions in the training corpus implies that the probability of the path will be estimated by zero. However, in the spline-based method, the surface of the spline associated with a path is not estimated by zero even though some transitions are absent in the training corpus, because the surface is expressed as a sum and not as a product of transition and emission probabilities.

To check if the choice of learning and test sets impacts system performance, we used the 10-folds cross validation test for spline and HMM based models. We did not use this test for the SVM-based model due to its performance limitations. We present in Table 5 the accuracy of each fold. We note that for each fold the spline based model surpasses the HMM based model, and on the other hand the 10 tests achieve performances close to average accuracy.

It remains to be noted that we did not consider it necessary to make comparisons with other disambiguation systems such as Madamira or Farasa since in [30] the authors compared Madamira with their lemmatization system based on HMMs, which is equivalent to the one used in Table 4, and they showed the superiority of the performances of their system.

5.3 Evaluation of the disambiguation system for the three tags

We will evaluate in this section our model on the three tags root, lemma and stem. The spline considered is the quadratic spline constructed from the (P1) stem weight estimate and the

Table 4.
Comparison between the three models.

	Quadratic Spline	Model based on HMM	SVM
Accuracy	94.15%	92.35%	88.43%
Speed	290 W/s	254 W/s	210 W/s

Fold	Accuracy of the model based on Spline	HMM
1	92.27%	91.83%
2	92.76%	91.86%
3	94.38%	94.09%
4	94.36%	94.03%
5	94.24%	94.05%
6	93.13%	91.94%
7	93.12%	93.02%
8	93.82%	92.30%
9	93.03%	93.01%
10	94.96%	90.37%
Average	**93.60%**	**92.65%**

Table 5. Results of the 10-folds cross validation test.

	Accuracy of the model based on Spline	HMM
Root	95.87%	94,48%
Lemma	94.98%	93,70%
Stem	94.15%	92,65%
All correct	92.43%	90,55%
All wrong	1.35%	1,93%

Table 6. Accuracies relating to the three tags.

(Tr2) right derivative estimate. We used the 10-folds cross validation test for spline and HMM based models. Table 6 presents the test results obtained for each tag analysed alone. Thus, lines 2, 3 and 4 present the average accuracy of all folds relating to each of the three tags stem, lemma and root respectively. The line ‘All correct’ displays the average percentage of words of the test corpus for which the three tags provided by the system coincide with those given by the annotators. Finally, the last line ‘All wrong’ displays the average percentage of tested words for which each tag provided by the system differs from that proposed by the annotators.

We observe that the performances of the system are very interesting. The difference in results for the three tags can be explained in part by the differences between the numbers of occurrences of each tag in the training corpus. Indeed, the occurrences of the roots are greater than those of the corresponding lemmas, which in turn are greater than those of the associated stems. As these occurrences are used in the training phase of the system (estimation of the Hermite data), the estimates of the weights of the roots are more precise than those of the lemmas, and the latter are more precise than those of stems. Moreover, by comparing the accuracies of the three tags with that of the last two lines of Table 6, we note that for almost all the tested words, the system tends to provide either three correct tags for each word or three wrong tags. Finally, the spline-based model performs better for the three tags than the HMM-based model.

6. Conclusion

In this paper we have presented an approach to identify the root, lemma and stem of words in Arabic language sentences. This approach is achieved in two steps. The analyser Alkhalil morpho sys is used in the first phase to identify for each word analysed out of context all its possible solutions (roots, lemmas and stems). Then, a disambiguation phase is carried out to identify the correct tag among these solutions. This phase is based on the use of spline functions.

The results obtained are very encouraging and we plan to improve them by using larger training corpus and using smoothing techniques to circumvent the problem of no transitions between word tags in the training corpus. We also intend to take advantage of the additional information provided by the analyser Alkhalil to better filter the transitions between the successive word tags. Finally, we envisage to develop a spline-based system that analyses the three tags together. A comparison between this system and the three presented in this paper will be made.

We are currently working on the exploitation of this approach to develop a POS Tag and a diacritisation system for the Arabic language. We have developed a demo for these three systems (light stemmer, lemmatizer and heavy stemmer) which can be consulted from the following address: http://demo.oujda-nlp-team.net/AlKhalil-Analyser/.

Once we finish developing the POS Tag and the diacritisation system, we intend to integrate them into the demo and then make the five systems open source.

Note

1 http://www.qamus.org/transliteration.htm Buckwalter transliteration:.

References

[1] I. Bounhas, N. Soudani, Y. Slimani, Building a morpho-semantic knowledge graph for Arabic information retrieval, Inf. Process. Manag. (2019) 102–124, http://dx.doi.org/10.1016/j.ipm.2019.102124.

[2] K.Z. Bousmaha, M.K. Rahmouni, B. Kouninef, L.B. Hadrich, A Hybrid Approach for the Morpho-Lexical Disambiguation of Arabic, J. Inf. Process. Syst. 12 (2016) 358–380, http://dx.doi.org/10.3745/JIPS.02.0041.

[3] S. Alkuhlani, N. Habash, A corpus for modeling morpho-syntactic agreement in Arabic: Gender, number and rationality, in: ACL-HLT 2011 – Proc. 49th Annu. Meet. Assoc. Comput. Linguist. Hum. Lang. Technol., 2011: pp. 357–362.

[4] M. Boudchiche, A. Mazroui, Evaluation of the ambiguity caused by the absence of diacritical marks in Arabic texts: Statistical study, in: 2015 5th Int. Conf. Inf. Commun. Technol. Access., IEEE, 2015: pp. 1–6. doi: 10.1109/ICTA.2015.7426904.

[5] N. Habash, Introduction to Arabic Natural Language Processing, Synth. Lect. Hum. Lang. Technol. 3 (2010) 1–187. doi: 10.2200/S00277ED1V01Y201008HLT010.

[6] E. Al-Shammari, J. Lin, A novel Arabic lemmatization algorithm, in: Proc. Second Work. Anal. Noisy Unstructured Text Data – '08, ACM Press, New York, New York, USA, 2008: pp. 113–118. https://doi.org/10.1145/1390749.1390767.

[7] R. Koulali, A. Meziane, Experiments with arabic topic detection, J. Theor. Appl. Inf. Technol. 50 (2013) 28–32, http://dx.doi.org/10.1007/978-3-642-25631-8_56.

[8] M. Boudchiche, A. Mazroui, M. Ould Abdallahi Ould Bebah, A. Lakhouaja, A. Boudlal, AlKhalil Morpho Sys 2: A robust Arabic morpho-syntactic analyzer, J. King Saud Univ. - Comput. Inf. Sci. 29 (2017) 141–146. doi: 10.1016/j.jksuci.2016.05.002.

[9] G. Farin, Curves and Surfaces for Computer-Aided Geometric Design, 3rd ed., Academic Press, Boston, 1993.

[10] S. Khoja, R. Garside, Stemming Arabic Text, United Kingdom Lancaster Univ. (1999).

[11] N. Yousef, A. Abu-Errub, A. Odeh, H. Khafajeh, An improved Arabic word's roots extraction method using N-gram technique, J. Comput. Sci. 10 (2014) 716–719, http://dx.doi.org/10.3844/jcssp.2014.716.719.

[12] A. Boudlal, R. Belahbib, A. Lakhouaja, A. Mazroui, A. Meziane, M. Bebah, A Markovian approach for arabic root extraction, Int. Arab J. Inf. Technol. 8 (2011) 91–98.

[13] L.S. Larkey, L. Ballesteros, M.E. Connell, Light Stemming for Arabic Information Retrieval, in: Arab. Comput. Morphol. Knowledge-Based Empir. Methods, Springer Netherlands, Dordrecht, 2007: pp. 221–243. https://doi.org/10.1007/978-1-4020-6046-5_12.

[14] L.S. Larkey, L. Ballesteros, M.E. Connell, Improving Stemming for Arabic Information Retrieval: Light Stemming and Co-occurrence Analysis, in: Proc. 25th Annu. Int. ACM SIGIR Conf. Res. Dev. Inf. Retr., ACM, 2002: pp. 275–282. https://doi.org/10.1145/564376.564425.

[15] M. Ababneh, R. Al-Shalabi, G. Kanaan, A. Al-Nobani, Building an effective rule-based light stemmer for Arabic language to improve search effectiveness, Int. Arab J. Inf. Technol. 9 (2012) 368–372.

[16] T. El-shishtawy, F. El-Ghannam, An accurate arabic root-based lemmatizer for information retrieval purposes, IJCSI, Int. J. Comput. Sci. Issues. (2012) 58–66.

[17] A. Mohammed, Z. Ayah, D. Mona, The Power of Language Music: Arabic Lemmatization through Patterns, in: M. Zock, A. Lenci, S. Evert (Eds.), Proc. 5th Work. Cogn. Asp. Lexicon, CogALex@COLING 2016, Osaka, Japan, December 12, 2016, The {COLING} 2016 Organizing Committee, 2016: pp. 40–50.

[18] A. Pasha, M. Al-badrashiny, M. Diab, A. El Kholy, R. Eskander, N. Habash, M. Pooleery, O. Rambow, R.M. Roth, MADAMIRA : A Fast, Comprehensive Tool for Morphological Analysis and Disambiguation of Arabic, in: Proc. 9th Lang. Resour. Eval. Conf., 2014: pp. 1094–1101.

[19] A. Shahrour, S. Khalifa, D. Taji, N. Habash, CamelParser: A system for Arabic Syntactic Analysis and Morphological Disambiguation, in: Proc. {COLING} 2016, 26th Int. Conf. Comput. Linguist. Syst. Demonstr., The COLING 2016 Organizing Committee, Osaka, Japan, 2016: pp. 228–232.

[20] I. Bounhas, R. Ayed, B. Elayeb, N. Bellamine Ben Saoud, A hybrid possibilistic approach for Arabic full morphological disambiguation, Data Knowl. Eng. 100 (2015) 240–254. doi: 10.1016/j.datak.2015.06.008.

[21] M. Nawar Ibrahim, M.N. Mahmoud, D.A. El-Reedy, Bel-Arabi: Advanced Arabic Grammar Analyzer, Int. J. Soc. Sci. Humanit. 6 (2016) 341–346. doi: 10.7763/IJSSH.2016.V6.669.

[22] K. Darwish, H. Mubarak, Farasa: A New Fast and Accurate {A}rabic Word Segmenter, in: Proc. Tenth Int. Conf. Lang. Resour. Eval., European Language Resources Association (ELRA), Portoro {ž}, Slovenia, 2016: pp. 1070–1074.

[23] A. Abdelali, K. Darwish, N. Durrani, H. Mubarak, Farasa: A Fast and Furious Segmenter for Arabic, in: Proc. 2016 Conf. North Am. Chapter Assoc. Comput. Linguist. Demonstr., Association for Computational Linguistics, 2016: pp. 11–16. https://doi.org/10.18653/v1/N16-3003.

[24] Y. Zhang, C. Li, R. Barzilay, K. Darwish, Randomized Greedy Inference for Joint Segmentation, POS Tagging and Dependency Parsing, in: Proc. 2015 Conf. North Am. Chapter Assoc. Comput. Linguist. Hum. Lang. Technol., Association for Computational Linguistics, 2015: pp. 42–52. https://doi.org/10.3115/v1/N15-1005.

[25] M. Attiya, M. Yaseen, K. Choukri, Specifications of the Arabic Written Corpus produced within the NEMLAR project, (2005). http://www.nemlar.org.

[26] C.D. Manning, H. Schütze, Foundations of Statistical Natural Language Processing, MIT Press, Cambridge, MA, USA, 1999. http://ics.upjs.sk/~pero/web/documents/pillar/Manning_Schuetze_StatisticalNLP.pdf.

[27] D. Neuhoff, The Viterbi algorithm as an aid in text recognition, IEEE Trans. Inf. Theory 21 (1975) 222–226, http://dx.doi.org/10.1109/TIT.1975.1055355.

[28] L.R. Rabiner, A tutorial on hidden Markov models and selected applications in speech recognition, in: Proc. IEEE, 1989: pp. 257–286. https://doi.org/10.1109/5.18626.

[29] N. Cristianini, J. Shawe-Taylor, An Introduction to Support Vector Machines and Other Kernel-based Learning Methods, Cambridge University Press, 2000. doi: 10.1017/CBO9780511801389.

[30] M. Boudchiche, A. Mazroui, A hybrid approach for Arabic lemmatization, Int. J. Speech Technol. 22 (2019) 563–573, http://dx.doi.org/10.1007/s10772-018-9528-3.

Corresponding author
Mohamed Boudchiche can be contacted at: moha.boudchiche@gmail.com