# Combinatorial Optimization Approach for Arabic Word Recognition Based on Adaptive Simulated Annealing

Zeineb Zouaoui(✉), Imen Ben Cheikh(✉), and Mohamed Jemni(✉)

Latice Laboratory, ENSIT, University of Tunis, Tunis, Tunisia
{zeineb.zouaoui,imen.becheikh}@gmail.com,
mohamed_jemni2000@yahoo.fr

**Abstract.** The present paper proposes an approach based on **a** combinatorial optimization technique for Arabic word recognition, distinguished by its flexional nature and significant topological variability. We treat a large vocabulary of Arabic decomposable words, which we choose to factorize them by their roots and schemes. We adopt a structure that resembles a molecular cloud. This design rhymes well with the Arabic linguistic philosophy of constructing words **from their roots.** Each sub-vocabulary, corresponding to a sub-cloud, embodies neighboring words, which are derived from one root and follow different schemes and forms of derivation, flexion, and agglutination (proclitic and enclitic). Therefore, we propose to use the metaheuristic simulated annealing (SA) method, as a recognition approach, in this wide cloud. It's an algorithm based on elastic comparisons between their structures and primitives. As an extension of previous works, we opt to implement the SA algorithm by integrating linguistic knowledge. Preliminary experiments were conducted on Arabic word corpus including samples and agglutinated words from APTI database and yielded interesting outcomes.

**Keywords:** Simulated annealing · Levenshtein distance · Morphological peculiarities · Combinatorial optimization · APTI database

## 1 Introduction

Arabic word recognition has become a very popular research area in recent years with a large number of possible applications (handwritten postal addresses, banks checks, etc.). The complexity of the recognition process depends on the script's type (printed or handwritten), the approach (holistic, pseudo-global and analytic) [1, 2] and the vocabulary size (reduced, large). Several approaches have been suggested, handling letters and/or pseudo-word levels, and numerous works have experienced statistical, neural, stochastic methods on different kinds of Arabic documents. Some works operated programmatically based on computation ability; others preferred to handle words cognitively while emulating human reading models.

In this work, we want, on one hand, to emphasize the fact that since humans see lexical and syntactic information in written words, it would be promising to underline linguistic knowledge integration in the recognition process. On the other hand, we aim


I. Nyström et al. (Eds.): CIARP 2019, LNCS 11896, pp. 480–489, 2019.
https://doi.org/10.1007/978-3-030-33904-3_45

to exploit combinatorial optimization, which had been proposed as a solution to artificial intelligence problems and test its efficiency in Arabic word recognition since Arabic is a morphologically complex and diverse language. Indeed, due to the fact that Arabic is a highly inflected and derived language, the number of effective forms go past 60 billion [3]. In this context, we intend to propose a novel approach based on a technique of combinatorial optimization that aims to recognize a large vocabulary of Arabic decomposable words (derived from roots), presenting different morphological aspects (derivational, inflectional and agglutinative).

This paper is organized as follows. Section 2 describes the major characteristics of the Arabic script. Section 3 is devoted to the description of the proposed approach as well as the lexicon organization. Then, the forth section reveals preliminary experiments and results to evaluate the approach. Finally, Sect. 5 summarizes the paper and proposes some perspectives.

## 2 Arabic Script Characteristics

### 2.1 Topological Specificities

Arabic script is written from right to left. Arabic writing is semi-cursive either in a printed or handwritten form. The Arabic alphabet contains 28 letters, which are primarily consonants. Most of them can change their forms according to their positions in the word; at the beginning, in the middle or at the end of the word. Some letters have four different forms like the letter (غـ,ـغـ,ـغ,غ) in Fig. 2. Furthermore, six letters have only two shapes, which are «ذ» (Thal), «د» (Dal), «ر» (Ra), «و» (Waw), «ز» (Zai) and «ا» (Alif). They are related to the letters that precede them, rather than to those that follow them. They form words composed of one or more parts called PAW (Peace of Arabic Word) or «Pseudo-Word» [16]. In addition, more than half of the Arabic characters include diacritics (one, two or three diacritical points) in their forms (see Fig. 1). These dots may be disposed either above or below the body of the letter, but they were never perceived simultaneously. The presence or the absence of these dots in their positions help to distinguish between letters that have exactly the same main shape.

| Feature | Designation | Description |
|---|---|---|
| A | Ascender | Ascending characteristic over the baseline |
| D | Descender | Descending characteristic below the baseline |
| L | Loop | Intersection between 2 closed boundaries |
| Da | Diacritics above | Diacritical points over the word body |
| Db | Diacritics below | Diacritical points below the word body |
| O | Otherwise | No primitives listed above |
| B,M,E,I | (B: beginning, M : Middle, E: End, I: Isolated) | Primitive position in the word |

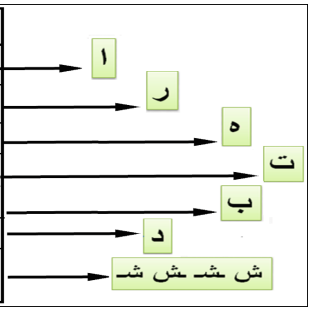


**Fig. 1.** Global features


**Fig. 2.** Different forms of the same Arabic letter

## 2.2 Morphological Specificities

The complexity and the richness of the Arabic script is attributed to the derivational structure of its morphological system. Most of the Arabic words are derived from roots by inserting prefixes, infixes, and suffixes. These words can be either conjugated verbs or specific names, such as machine names, lawsuit names, time and place names, preference nouns, and analogue adjectives. They are known as decomposable words [2]. Non-decomposable words, however, correspond for instance to the names of countries, numbers, etc. This diversity is due to Arabic derivational aspect, which consists of the fact that Arabic words are derived from a root according to a given pattern (scheme). This procedure allows the generation of new words by adding prefixes that precede a base (is the set of root consonants), by injecting infixes that appear between the basic letters and by adding suffixes coming at the end. The product of the derivation of a root following a pattern is called radical. This morphological richness appears, also, on the inflectional plan. Indeed, a radical undergoes inflected conjugations that can change the prefixes and suffixes and add additional letters. In this way, the elements of derivational and inflectional conjugations of Arabic comprise the root, scheme, personal pronoun, time, gender, number, function, etc. Finally, Arabic is also characterized by its agglutinative morphology. There are two types of agglutination: proclitic and enclitics [5]. We designate simple proclitic (morphemes of one letter) and compound proclitic (morphemes of several letters). The first are coordinating, conjunctions, prepositions and the latter are obtained by combining first ones. The enclitic is a supplement pronoun that can be single or double and is attached to the word that precedes it. Figure 3 shows the example of the word «سيراجعونه» that derives from the root «رجع», conjugated according to the scheme «فاعل » with the pronoun « هم» (they: plural masculine) and adding the proclitic «س» (designating future tense) and the enclitic «ه» (designating the object complement: him). Furthermore, a single Arabic word can substitute a completely English sentence; as the word «أسيراجعونه» in Fig. 4, can be translated to the sentence (**Will they review it?**).

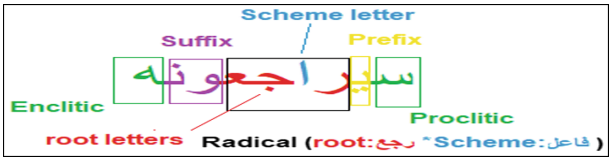


**Fig. 3.** Morphems of Arabic word


**Fig. 4.** Sample of agglutinated Arabic word

## 3 Approach Proposal

### 3.1 Motivation

Our work tries to benefit from the combinatorial optimization techniques to solve pattern recognition problems known for their strongly exponential complexity. These techniques had been generally proposed for solving artificial intelligence problems in large solution spaces. Thus, we attempt to evaluate how these techniques could be generally effective with word recognition and especially with Arabic one. We focused especially on SA algorithm. It is a powerful probabilistic metaheuristic algorithm originally proposed by Metropolis et al. [6]. This method has proven to be quite versatile and efficient for wide range of combinatorial optimization problems. For example, in the travelling Salesman problem, experiments proved that SA becomes efficacious above 800 cities [7]. In addition, it seems robust with data mining applications particularly when clustering large categorical data sets [8]. Furthermore, it has been proven efficient for partitioning graph problem, pattern detection of seismic applications and job-shop scheduling, etc. [9, 10].

Therefore, SA method looks highly effective in exploring and exploiting the large search space associated with problems having particular properties [11]. Why is the writing recognition problem, characterized by its wide variability and particular peculiarities, not highlighted? (1) So, topologically, we opt to adjust the SA elastic comparison by calculating the Levenshtein (or editing) distance. It is a metric method suggested by Levenshtein [6], to compare two words, two strings, and more generally, two sequences. The edit distance is defined as the minimum number of operations to convert the first word into the second word, by taking into account the presence of only three possible operations:

- replacing a character with another,
- deleting a character,
- inserting a character.

Edit distance algorithm is simple to code in a short calculation time. Hence, several studies have tested edit distance with structural methods of pattern recognition [12, 8 and 13]. Carbonnel in [14] presented an automatic learning method to calculate the edit distance in order to adapt it to online handwriting. Moreover, Gueddah in [13] proposed to integrate a measurement of proximity and similarity between Arabic characters in Levenshtein algorithm to better suggest and schedule automatic correction of spelling errors detected in typed Arabic documents. He achieved interesting outcomes. Inspired by these studies, we thought of adjusting this elastic comparison algorithm to

Arabic word recognition in order to privilege the global over the local and to absorb the writing variability.

(2) Morphologically, we attempt to integrate Arabic linguistic peculiarities in SA neighborhood. It can deeply expand the performance and the accuracy of SA [14]. We have chosen to organize the SA neighborhood around morpho-phonological concepts (root, prefix, suffix, enclitic, proclitic, etc.).

### 3.2 Arabic Vocabulary as Molecular Cloud

Recall that we have decided to organize our vocabulary of decomposable words as a molecular cloud, whose organization emulates the Arabic linguistic philosophy of word construction around the roots. Each sub-cloud assembles neighboring words derived from the same root according to multiple derivational forms, such as flexion and agglutination (proclitic and enclitic) (see Figs. 5 and 6). This factorization highlighted common morphological entities, like roots, schemes, with various conjugating elements (e.g. present, past, singular, dual, plural, masculine, feminine, etc.) and agglutination, etc. Figure 5 gives an overview of the sub-cloud that represents the root « بعد » (i.e. to go far) linked to its derivatives, which are presented as molecules. On the other hand, Fig. 6 represents the union of several sub-clouds (black balls represent sub-cloud nuclei), which relatively correspond to the roots, in order to constitute the global molecular cloud.

This morpheme-based structuring model ensures the learning phase that characterizes the majority of pattern recognition systems. In a first step, we have prepared an initial cloud of morphologically related words enriched by diverse rules of verb conjugations in different tenses, with various persons, flexion, pronouns, and functions. In the second step, we have integrated some morphological instructions in order to connect the generated words and sub-clouds. Finally, the first obtained cloud represents the initial solution space that is automatically stabilized in accordance with SA iterations itself.

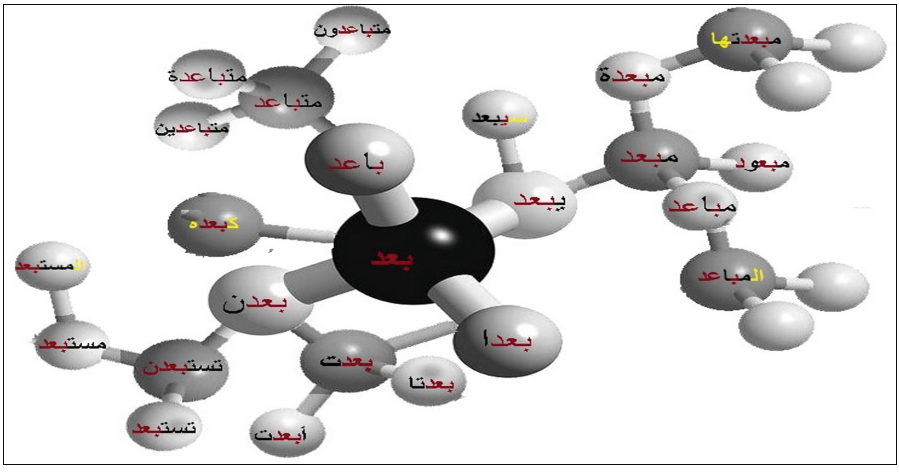


**Fig. 5.** Part of the sub-cloud of the root « بعد » (i.e. to go far)

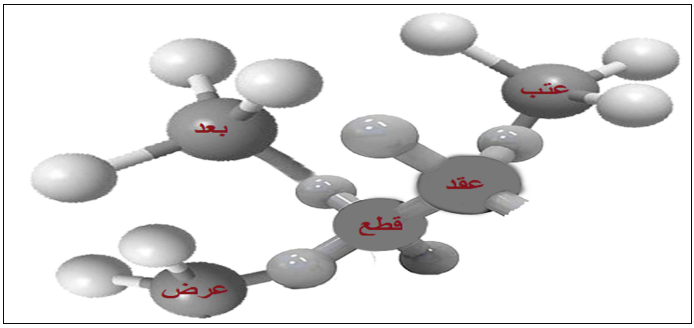


**Fig. 6.** Molecular cloud: union of various sub-clouds (black balls represent sub-cloud nuclei)

### 3.3 Recognition

As shown above, we have opted for SA metaheuristic method, whose major advantage is to escape local optima and converge to the global optima (see algorithm in Fig. 7). To adapt the SA algorithm to a particular problem, we must specify the state space, the temperature, the neighbor selection method, the probability transition function and the annealing schedule. These choices have a considerable effect on the results [5]. The chosen parameters were then updated or not based on the decision rule of our SA algorithm.

To handle our open vocabulary, we adopted structures of primitive sequences (e.g. LB AE denote the loop at the beginning followed by an ascender at the end (see Fig. 1)) extracted from the word images in previous works [15]. Therefore, we chose Levenshtein comparative method, between words and their features, that operates as follows:

We look for a recursive solution by introducing ED(i, j) the optimal distance between the first i characters of the first string and the first j characters of the second string. We deduce initial values:

$$\begin{aligned} \mathrm{ED}\ (0,0) &= 0 \\ \mathrm{ED}(0,\mathrm{i}) = \mathrm{ED}(\mathrm{i},0) &= \mathrm{i}, \qquad \text{and the recursive relationship:} \\ \mathrm{ED}\ (\mathrm{i},\mathrm{j}) = \mathrm{Minimum}\{&\mathrm{ED}\ (\mathrm{i}-1,\mathrm{j})+1, \qquad //\text{deletion} \\ &\mathrm{ED}\ (\mathrm{i},\mathrm{j}-1)+1, \qquad // \text{ insertion} \\ &\mathrm{ED}\ (\mathrm{i}-1,\mathrm{j}-1)+\mathrm{cost}\ \} \qquad // \text{ permutation} \end{aligned}$$

$$\text{With } cost = \begin{cases} 0 & if\ \ c(i) = c(j) \\ 1 & else \end{cases}$$

To reach the ED(i, j) point, you have to insert a letter, delete a letter, replace a letter, or do nothing if the two characters c (i) and c (j) are equal.

Table 1 shows an example of the Levenshtein calculation between the word «يبعد» and the word «يباعد», the edit distance is equal to 2.

**Table 1.** An example of edit distance calculation

| | | ي | ب | ع | د |
|---|---|---|---|---|---|
| | 0 | 1 | 2 | 3 | 4 |
| ي | 1 | 0 | 1 | 2 | 3 |
| ب | 2 | 1 | 0 | 1 | 2 |
| ا | 3 | 2 | 1 | 1 | 2 |
| ع | 4 | 3 | 2 | 2 | 2 |
| د | 5 | 4 | 3 | 3 | 2 |

Until then, the calculation of the edit distance is classic (Classic ED). It deals with primitives as normal strings, i.e. it does not differentiate between the root consonant or scheme, and the conjugated letters that appear in prefixes and suffixes. Therefore, this classic ED calculation method can cause confusion, when we choose the optimal solution. Indeed, as mentioned in Table 2 below, the edit distance between the word "يبعد" derived from the scheme "فعل " and all the proposed solutions, which are derived from different schemes, is the same.

To remedy this problem, we propose in this work to integrate a new cost in the editing operations, which enhance the presence or the absence of scheme letters (ShL). This cost differs depending on letter position schemes; whether it is in the ideal location or not. The pseudo-code is as follow:

$$\begin{aligned} \text{EDsh}\ (0,0) &= 0 \\ \text{EDsh}\ (0,\mathrm{i}) = \text{EDsh}(\mathrm{i},0) &= \mathrm{i} \qquad \text{and the recursive relationship:} \\ \text{EDsh}\ (\mathrm{i},\mathrm{j}) = \text{Minimum}\ \{\ &\text{EDsh}\ (\mathrm{i}-1,\mathrm{j}) + 1, \qquad // \text{ deletion} \\ &\text{EDsh}\ (\mathrm{i},\mathrm{j}-1) + \text{Cost1}, \qquad // \text{ insertion} \\ &\text{EDsh}\ (\mathrm{i}-1,\mathrm{j}-1) + \text{Cost2}\} \qquad // \text{ permutation} \end{aligned}$$

With

$$\text{Cost 1} = \begin{cases} 1.7 & \text{if } (\mathrm{c}(\mathrm{i}) \text{ is not a ShL}) \\ 0.3 & \text{else} \end{cases}$$

$$\text{Cost 2} = \begin{cases} 0 & \text{if } (\mathrm{c}(\mathrm{i}) \text{ is not a ShL } \&\&\ \mathrm{c}(\mathrm{i}) = \mathrm{c}(\mathrm{j})) \\ 0.2 & \text{if } (\mathrm{c}(\mathrm{i}) \text{ is a ShL } \&\&\ \mathrm{c}(\mathrm{i}) = \mathrm{c}(\mathrm{j}) \ \&\&\ \mathrm{i} = \mathrm{j}) \\ 0.5 & \text{if } (\mathrm{c}(\mathrm{i}) \text{ is a ShL } \&\&\ \mathrm{c}(\mathrm{i}) = \mathrm{c}(\mathrm{j}) \ \&\&\ \mathrm{i} \neq \mathrm{j}) \\ 0.3 & \text{if } (\mathrm{c}(\mathrm{i}) \text{ is not a ShL } \&\&\ \mathrm{c}(\mathrm{i}) \neq \mathrm{c}(\mathrm{j})) \\ 1.7 & \text{if } (\mathrm{c}(\mathrm{i}) \text{ is a ShL } \&\&\ \mathrm{c}(\mathrm{i}) = \mathrm{c}(\mathrm{j})) \end{cases}$$

$$\text{NewED}\ (\mathrm{i},\mathrm{j}) = \text{ED}\ (\mathrm{i},\mathrm{j}) + \text{EDsh}(\mathrm{i},\mathrm{j})$$

**Table 2.** Samples of proposed solutions with the new edit distance calculation

| Word to recognize | Solutions | Scheme | Classic ED | New ED |
|---|---|---|---|---|
| يبعد<br>فعل | يبتعد | افتعل | 1 | 2,9 |
| | مبعد | مفعل | 1 | 2,6 |
| | يبعده | فعل | 1 | 1,3 |

```
Current-solution = W =select a random word from the cloud
NewED(w)= New Edit distance between the unknown word and
W (current-solution)
Set the initial temperature initial_T
While (temperature >0 and non-convergence)
Neighbor = select the best neighborhood word solution
Calculate delta = NewED (neighbor) – NewED(current-
solution)
If (delta <= 0)
  Current-solution = neighbor
Else
  Select new neighbor with probability e^-(delta/t)
End
Decrease the temperature
End
Output the final solution
```

**Fig. 7.** Simulated annealing algorithm with a new Edit distance

## 4 Preliminary Experimental Results

The proposed approach has been tested using a database of printed Arabic word images in different fonts and sizes derived from tri-consonantal healthy roots (see Fig. 8), which embody samples extracted from APTI database [16].

Our training corpus is a set of structural primitives that we extract from Arabic word images. The extraction process is expanded in the previous work [15]. We handled 3200 samples derived from 41 tri-consonantal roots by following different schemes. They are also enriched by a variety of agglutinative and flexional features.

Experimental tests showed that the highest recognition rates (99.84%) are achieved at a temperature equal to 100 with stable cooling equal to 0.9. Furthermore, Fig. 9 demonstrates that the higher the cooling is, the more important the recognition rate is, with initial temperature fixed at 100 (for small response time).


**Fig. 8.** Word samples extracted from APTI Database

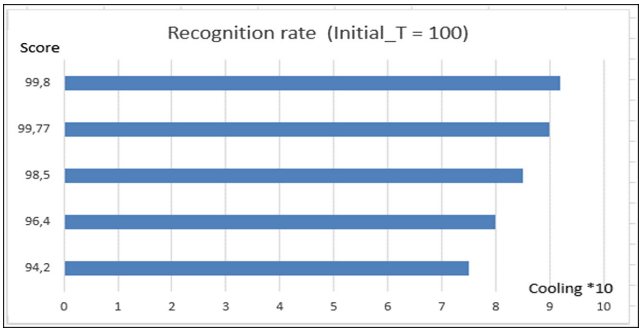


**Fig. 9.** Evolution of the recognition rate according to cooling

Moreover, the problem of long stagnation in local wells, that we may encounter while using Levenshtein distance, is resolved due to the integration of new costs in the edit distance calculating.

## 5 Conclusion

To summarize, in this work we proposed a novel approach based on combinatorial optimization and precisely simulated annealing algorithm for the recognition of wide vocabulary of decomposable Arabic words. We chose to integrate linguistic knowledge in both vocabulary structure and the recognition phase through the SA algorithm. In fact, we have organized our vocabulary as a molecular cloud, whose organization matches the Arabic philosophy of constructing words around roots. Each sub-cloud includes neighboring words derived from one root. Preliminary experimentation has been held on a morphological cloud of more than 1000 nodes structured on 41 sub-clouds that correspond to 41 tri-consonantal Arabic roots. We have used simulated annealing on structural primitive vectors extracted from word images including APTI samples. We started by classic elastic comparison. Initial results are promising but with stagnation limit in local wells. To remedy this weakness, we have integrated linguistic knowledge in the edit distance calculation and we have precisely highlighted the

scheme letters to help the SA algorithm to achieve the optimal solution, which corresponds to the vocabulary structure.

As perspectives, first, we will try to extend our training corpus in order to generate a very wide molecular morphological cloud (MMC). Second, we propose to integrate a whole model (neural or markovian) in each node, so that a node would be able to neither calculate an ED nor take part in the root research solution within the eventually enormous cloud.

## References


1. Touj, S., Ben Amara, N., Amiri, H.: A hybrid approach for off-line Arabic handwriting recognition based on a planar Hidden Markov Modeling. In: ICDAR 2007, Brazil, pp. 964–968 (2007)
2. Avila, J.M.: Optimisation de modèles markoviens pour la reconnaissance de l'écrit. Ph.D., University of Rouen (1996)
3. Cheriet, M., Beldjehem, M.: Visual processing of Arabic handwriting: challenges and new directions. In: SACH 2006, India, pp. 1–21 (2006)
4. Kanoun, S., Alimi, A.M., Lecourtier, Y.: Natural language morphology integration in off-line Arabic optical text recognition. IEEE Trans. Syst. Man Cybern.—Part B: Cybern. **41**(2), 579–590 (2011)
5. Ben Cheikh, I., Allagui, I.: Planar Markovian approach for the recognition of a wide vocabulary of Arabic decomposable words. In: ICDAR, pp. 1031–1035 (2015)
6. Levenshtein, V.: Binary codes capable of correcting deletions, insertions and reversals. SOL Phys. Dokl. **10**, 707–710 (1966)
7. Anigbogu, J.: Reconnaissance de textes imprimés multifontes àl'aide de modèles stochastiques et métriques. Ph.D., University of Nancy 1 (1992)
8. Rani, S., Singh, J.: Enhancing Levenshtein's edit distance algorithm for evaluating document similarity. In: Sharma, R., Mantri, A., Dua, S. (eds.) ICAN 2017. CCIS, vol. 805, pp. 72–80. Springer, Singapore (2018). https://doi.org/10.1007/978-981-13-0755-3_6
9. Huang, K., Hsieh, Y.: Very fast simulated annealing for pattern detection and seismic. In: IEEE International Geoscience and Remote Sensing Symposium, IGARSS 2011, Vancouver, BC, Canada, 24–29 July 2011
10. Goyal, A., Sourav, P.A., Thangavelu, A.: A comparative analysis of simulated annealing based intuitionistic fuzzy k-mode algorithm for clustering categorical data. Int. J. Comput. Inf. Syst. Ind. Manag. Appl. **9**, 232–240 (2017)
11. Baptiste, A.: Les métaheuristiques en optimisation combinatoire. Thesis to obtain probative exam on computing, National Conservatory of Arts and Crafts, Paris (2006)
12. Gueddah, H.: La correction orthographique des textes arabes: Contribution àla résolution d'ordonnancement et de l'insuffisance des lexiques. Ph.D. University of Mohamed V, Rabat (2017)
13. Carbonnel, S., Anquetil, E.: Apprentissage automatique d'une distance d'édition dédié à la reconnaissance d'écriture manuscrite. In: CIFED (2004)
14. Xinchao, Z.: Simulated annealing algorithm with adaptive neighborhood. Appl. Soft Comput. **11**, 1827–1836 (2011)
15. Ben Cheikh, I., Zouaoui, Z.: HMM based classifier for the recognition of roots of a large canonical Arabic vocabulary. In: ICPRAM, pp. 244–252 (2013)
16. https://diuf.unifr.ch/diva/APTI/