Pattern Recognition Letters 36 (2014) 74–80

Contents lists available at ScienceDirect

Pattern Recognition Letters

journal homepage: www.elsevier.com/locate/patrec

A new iterative algorithm for computing a quality approximate median of strings based on edit operations

J. Abreu [a,*], J.R. Rico-Juan [b]

[a] Dpto Informática, Universidad de Matanzas, Carretera a Varadero Km. 3 1/2, Matanzas, Cuba
[b] Dpto Lenguajes y Sistemas Informáticos, Universidad de Alicante, San Vicente del Raspeig, Alicante, Spain

ARTICLE INFO

Article history:
Received 2 October 2012
Available online 3 October 2013

Communicated by R. Davies

Keywords:
Approximate median string
Edit distance
Edit operations

ABSTRACT

This paper presents a new algorithm that can be used to compute an approximation to the median of a set of strings. The approximate median is obtained through the successive improvements of a partial solution. The edit distance from the partial solution to all the strings in the set is computed in each iteration, thus accounting for the frequency of each of the edit operations in all the positions of the approximate median. A goodness index for edit operations is later computed by multiplying their frequency by the cost. Each operation is tested, starting from that with the highest index, in order to verify whether applying it to the partial solution leads to an improvement. If successful, a new iteration begins from the new approximate median. The algorithm finishes when all the operations have been examined without a better solution being found. Comparative experiments involving Freeman chain codes encoding 2D shapes and the Copenhagen chromosome database show that the quality of the approximate median string is similar to benchmark approaches but achieves a much faster convergence.

1. Introduction

Extending the concept of "median" to structural representations such as strings has been a challenging issue in Pattern Recognition for some time, as it is shown in the review presented in Jiang et al. (2004). This problem arises in many applications such as 2D shape representation and prototype construction (Jiang et al., 2000; Bunke et al., 2002), the clustering of strings (Lourenço and Fred, 2005), Self-Organized Maps of strings (Kohonen, 1998; Fischer and Zell, 2000) or the combination of multiple source translations (González-Rubio and Casacuberta, 2010).

Formally, given a set $S = \{S_1, S_2, \ldots, S_n\}$ of strings over the alphabet $\sum$ and a distance function $D(S_i, S_j)$ which measures the dissimilarity between strings S_i and S_j, the distance from a string S' to all the strings in S can be computed by the expression (1).

$$SOD(S') = \sum_{S_i \in S} D(S', S_i) \tag{1}$$

The *median string* is the string $\widehat{S} \in \sum^*$ that minimizes (1). This string is also denoted as the *generalized median string*. A common approximation to the true median string is the *set median*, a string in S which minimizes (1). It is not necessary for either the median string or the set median to be unique.

An exact algorithm to compute the median of a set of strings was proposed by Kruskal (1983). However, in most practical applications this is not a suitable approach due to the high computational time requirements. As Casacuberta and Antonio (1997) and Nicolas and Rivals (2005) pointed out, there are various formulations of this problem within the NP-Complete class. Several approximations have therefore been proposed. One approach that has been studied by several authors is that of building the approximate median by using the successive changes of an initial string. One or more pertubations can be applied at a time, as in the works of Martínez-Hinarejos et al. (2003) and Fischer and Zell (2000), respectively. The results of empirical testing show that the first approach leads to high quality approximations but requires more computational time. The principal motivation of this work is to describe a new algorithm able to compute a quality approximation to the median string like that of Martínez-Hinarejos et al. (2003), but requires significantly less computational effort. In Section 2 some related works are examined. Section 3 describes the proposed approach and provides an analysis of the computational cost bounds for the algorithm. Various comparative experiments are described in Section 4. Finally, Section 5 shows our conclusions and some lines for further research.

2. Related works

Many approximate solutions have been described since Kruskal (1983) proposed an exact algorithm that could be used to compute

* Corresponding author. Fax: +34 965909326.
E-mail addresses: jose.abreu@umcc.cu (J. Abreu), JuanRamonRico@ua.es (J.R. Rico-Juan).

0167-8655/$ - see front matter © 2013 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.patrec.2013.09.014

the median string for a given set S of N strings of a length of l and the Levenshtein (Levenshtein, 1966) metric. This algorithm runs in $\mathcal{O}(l^N)$ proportional time. A number of heuristics therefore address this difficulty by reducing the size of the search space. Some authors, such as Olivares and Oncina (2008), have studied the approximation to the median string not only under the Levenshtein edit distance but also under the stochastic edit distance (Ristad and Yianilos, 1998). In other works, the search for the approximate median is not performed directly in the string space but in a vectorial space in which the strings are embedded; this is the approach studied in Jiang et al. (2012) which also relies on the weighted median concept described by Bunke et al. (2002).

One general strategy is to construct the approximate median letter by letter from an initial empty string. It is necessary to define a goodness function to decide which symbol is the next to be appended. The greedy procedure described in Casacuberta and Antonio (1997) implements this approach. An improvement to the aforementioned method is described in Kruzslicz (1999) through the use of a refined criterion which allows the next letter to be selected. Another approach that has been studied by several authors is that of building the approximate median by using successive perturbations of an initial string. Two important issues regarding this kind of method are; how to select a perturbation leading to an improvement and how to make the algorithm converge faster without spoiling the results. Another interesting topic is that of studying the effect of performing modifications one by one or simultaneously. Kohonen (1985) starts from the set median and systematically changes the guess string by applying insertions, deletions and substitutions in every position. In Martínez-Hinarejos et al. (2003) the authors propose to improve a partial solution $\widehat{S}$ generating new candidates by applying all possible substitutions, insertions or deleting the symbol at a position i. The new partial solution is the string, selected from all the new candidates and $\widehat{S}$, which minimizes (1). This procedure is repeated for every position i. The effect of choosing a different initial string as the set median or a greedy approximation is also studied. Theoretical and empirical results show that this method is capable of achieving very good approximations to the true median string. Note that these methods do not define a criterion to compare the operations in order to select which one can lead to better results in each case. In Martínez-Hinarejos et al. (2002) authors describe alternatives to speed up the computation of the approximated median string. Based on information provided by the weight matrix used to compute the edit distance, certain operations are preferred instead of others. For example, not all possible substitutions are tested but only the two closest symbols to the one in the analysed position.

Some heuristic knowledge that can help to assess how promising a modification will be are included in Fischer and Zell (2000) and Mollineda (2004). The quality of a partial solution $\widehat{S}$ is evaluated by computing its distance from every string in the set. Thus, it is also possible to discover the sequences of edit operations. In an attempt to speed up the convergence of the search procedure, these authors propose the simultaneous performance of several modifications by applying the most frequent edit operation, including "do nothing" in each position of the partial solution. This process is repeated while modifications increase the quality of the partial solution.

This approach has two potential drawbacks: applying the most common operation in every position does not guarantee the best results and although it might be relatively simple to figure out how applying just one operation will affect $SOD(\widehat{S})$, this does not hold when several changes are made at the same time. For example, let $\widehat{S}$ be a partial solution and op_i be an edit operation which occurs several times when computing the distance from a partial solution to strings in S. Op_i thus determines a subset $S^{YES} \subseteq S$ of those strings in which op_i occurs when computing the distance from $\widehat{S}$. There is also another set $S^{NO} = S - S^{YES}$. Let $\widehat{S}'$ be a new solution after applying op_i to $\widehat{S}$. Intuitively, it may be expected that the distance from $\widehat{S}'$ to strings in S^{YES} decreases regarding $\widehat{S}$. A formal discussion of this result can be found in Bunke et al. (2002). The effect on the strings in S^{NO} clearly needs to be taken into account. Since sets induced by each operation may be different when applying multiple operations, it might be very difficult to characterize the effect on $SOD(\widehat{S})$. Empirical results, which will be discussed later, suggest that those methods that apply multiple perturbations at the same time are able to find a better approximation to the median quickly. However, approaches which perform modifications one by one, such as Martínez-Hinarejos et al. (2003), significantly outperform the former methods with respect to the average distance to the set of the approximate median computed.

Table 1
Computation of the edit distance cost from $\widehat{S}^t = \{5,5,0\}$ to $S_1 = \{3,1,1,2\}$ and $S_2 = \{0,6,1,6\}$. Substitutions of a symbol a by a symbol b have cost $min\{|a-b|, 8-|a-b|\}$ while deletions and insertions have cost of 2. An optimal path is shaded in order to follow the best cost operations easily and visually.

(a)

		3	1	1	2
	0	2	4	6	8
5	2	2	4	6	8
5	4	4	6	8	9
0	6	6	5	7	9

(b)

		0	6	1	6
	0	2	4	6	8
5	2	3	3	5	7
5	4	5	4	6	6
0	6	4	6	5	7

3. A new algorithm for computing a quality approximate median string

As noted earlier, a general scheme that can be used to search for an approximate median string is:

- select an initial coarse approximation to the median as the set median.
- generate a new solution by performing some modifications to the current solution.
- repeat while a particular modification leads to an improvement or another stop condition holds.

The works commented on Section 2 suggest that when it is necessary to find a quality approximation to the median string, applying modifications one by one would appear to be a better strategy. The theoretical results in Jiang and Bunke (2002) and Martínez-Hinarejos (2003) show that the approximation computed by the algorithm proposed in Martínez-Hinarejos et al. (2003) is very close to the lower bound obtained for the value of $SOD(\widehat{S})$ for the true median.

3.1. Computing the approximate median string

The algorithm in Martínez-Hinarejos et al. (2003) tests every possible operation in each position of the partial solution and it might therefore be very useful to study how to reduce the size of the search space without spoiling the quality of results, which is one of the principal motivations of this work. The proposed algorithm is based on two main ideas:

- selecting the appropriate modification by paying attention to certain statistics from the computation of the edit distance from the partial solution to every string in the set.
- applying modifications one by one.

Heuristic information could help to avoid testing a number of useless solutions, which would reduce the amount of times that