A Stochastic Approach to Median String Computation

Cristian Olivares-Rodríguez and Jose Oncina

Departamento de lenguajes y sistemas informáticos,
Universidad de Alicante
{colivares,oncina}@dlsi.ua.es

Abstract. Due to its robustness to outliers, many Pattern Recognition algorithms use the *median* as a representative of a set of points. A special case arises in Syntactical Pattern Recognition when the points (prototypes) are represented by strings. However, when the edit distance is used, finding the median becomes a NP-Hard problem. Then, either the search is restricted to strings in the data (*set-median*) or some heuristic approach is applied. In this work we use the (conditional) stochastic edit distance instead of the plain edit distance. It is not yet known if in this case the problem is also NP-Hard so an approximation algorithm is described. The algorithm is based on the extension of the string structure to multistrings (strings of stochastic vectors where each element represents the probability of each symbol) to allow the use of the Expectation Maximization technique. We carry out some experiments over a chromosomes corpus to check the efficiency of the algorithm.

Keywords: Median String, (Multi)string, Stochastic Edit Distance.

1 Introduction

Given a set of data points and a dissimilarity function, the *median* (also called *generalized median* or *geometric median*) point is defined as the point, in the whole space of the data points, that minimizes the sum of the dissimilarities with respect to the points of the set. It is easy to see that the customary definition used in statistics is a special case of this definition when the points are real numbers and the dissimilarity function is the absolute value of their differences. In a similar way, the mean can be defined as the point that minimizes the sum of the squared dissimilarities.

Due to the fact that the median is less sensitive to outliers than the mean (because of the squares), many Pattern Recognition techniques relies in computing the median instead of the mean when robustness is required. A special and very common case in Syntactical Pattern Recognition arises when the points are strings of features (symbols) and the dissimilarity function is the edit distance [1]. Unfortunately, in this case, the problem becomes NP-Hard [2]. Then, although some effectiveness is lost, it is customary to recur to the so called *set median* [3], that is, the median is restricted to be a point in the data set.

N. da Vitora Lobo et al. (Eds.): SSPR&SPR 2008, LNCS 5342, pp. 431–440, 2008.

Kohonen, in 1985 [4], proposed an approximation algorithm to find the generalized median based on a process of perturbation of the set median in order to modify it in the direction of the generalized median string. This results was improved in 2000 by Martínez-Hinarejos *et al.* [5][6]. Moreover, their experiments showed that using the generalized median instead of the set median improved their results.

In this work we are going to study the special case when the dissimilarity measure is the stochastic edit distance [7][8]. This distance is a reinterpretation of the classical edit distance where the edition costs are interpreted as the probabilities of inserting, deleting or substituting a symbol. The stochastic distance is then the minus logarithm of the probability that a string is a transformation of the other. The stochastic edit distance has the quality that the problem of learning the costs can be stated as a maximum likelihood problem and then, the Expectation Maximization technique [9] can be used [7] [8]. Unfortunately, although it is an open problem there are strong evidences that the problem of finding the median using the stochastic edit distance is also an NP-Hard problem.

In this work we propose an approximation algorithm to find the median. The proposed algorithm relies on an extension of the string data structure (a concatenation of symbols), to the multistring data structure, (a concatenation of stochastic vectors where each element represents the probability of each symbol). This pass to the continuum in the symbol space is seized to apply the expectation maximization technique and find the multistring that minimizes the sum of the distances to the set. After that, two possibilities are offered: the first is to recover a string by thresholding the multistring and the second is to transform the Pattern Recognition algorithm that uses the median to use multistrings instead of strings. In this paper we are going to explore the second one.

In the experimental section, since, to our knowledge, no other technique to find the generalized median string exists, our algorithm is compared with a straight forward adaptation of the Martínez-Hinarejos one.

2 Stochastic String Edit Distance

Let $X = \{a_1, \ldots, a_n\}$ be a finite set (*alphabet*). A string x is any finite concatenation of symbols in X. Let X^* denote the set of all the strings that can be made using symbols in X and X^n the set of all the strings with exactly n symbols. The empty string is represented by ϵ. Let x be a string, x_i denotes the i-th symbol of x, then $x = x_1 \ldots x_m$ where $m = |x|$. On the following we are going to use $a, b, \ldots$ to denote symbols and $x, y, \ldots$ to denote strings.

Classic String Edit Distance [1] is a dissimilarity measure between strings defined as the minimum number of *edit operations* needed to transform one string into the other, where an edit operation is an *insertion*, *deletion*, or *substitution* of a single symbol. This distance can be extended to use edit operations costs instead of simple edit operation counts. The edit distance can also be viewed as a model of the modifications suffered by a sequence of symbols when traversing a noisy channel.

More formally, let X (Y) be the alphabet of the input (output) strings. Let $E_d = \{(a, \epsilon) : a \in X\}$, $E_i = \{(\epsilon, b) : b \in Y\}$, $E_s = \{(a, b) : a \in X, b \in Y\}$ represent the deletion, insertion and substitution edit operations, and let $E = E_i \cup E_s \cup E_d$. The string Edit Distance is defined by a triple (X, Y, c_e) where $c_e : E \to \mathbf{R}$ is the primitive cost function. This structure induces a dissimilarity function d over pairs of strings $d : X^* \times Y^* \to \mathbf{R}$ as:

$$d(x,y) = \min \begin{cases} [c_e(a,b) + d(x',y')]_{x=x'a \wedge y=y'b} \\ [c_e(a,\epsilon) + d(x',y)]_{x=x'a} \\ [c_e(\epsilon,b) + d(x,y')]_{y=y'b} \end{cases} \tag{1}$$

Where $[x]_p$ returns x if predicate p is true and zero otherwise. (Note that $d(x, y)$ can be computed in $O(|x| \cdot |y|)$ time using dynamic programming techniques.)

Following the noisy channel model, if it is assumed the edit operations are based on a random phenomenon, a dissimilarity edit distance like measure can be defined as the probability of having string y in the output provided string x is in the input of the channel $(p(y|x))$[8][1]

Suppose the edit operations are independent. Let $c : E \to \mathbf{R}$ the cost function where:

- $c(a, \epsilon)$ is interpreted as the probability of deleting the symbol a provided the symbol a is the next symbol in the input string.
- $c(\epsilon, b)$ is interpreted as the probability of inserting the symbol b
- $c(a, b)$ is interpreted as the probability of substituting the symbol a by b provided the symbol a is the next symbol in the input string.

In this framework, a new probability should be introduced to represent the probability of stopping making insertions at the end of the string. Let we call γ this probability.

Then the probability of obtaining the string y provided the string x is in the input of the noisy channel, is the sum of the probabilities of all the possible ways of transforming the string x into y.

More formally, given a pair of strings $(x, y) \in X \times Y$, we denote by $E(x, y)$ the set of all the edit operation sequences that can transform x into y, that is $E(x, y) = \{(x_1, y_1) \dots (x_n, y_n) : (x_i, y_i) \in E, x_1 \dots x_n = x, y_1 \dots y_n = y\}$.

Let $z = (x_1, y_1) \dots (x_n, y_n)$ be a sequence of edit operations, we define

$$p(z) = \prod_{i=1}^{n} c(x_i, y_i)$$

the probability of generating y given x is defined as:

$$p(y|x) = \sum_{z \in E(x,y)} p(z)\gamma$$

[1] A similar approach was previously used by Ristad *et al.* [7] but based in a joint probability distribution. The results in this paper can be easily extended to the Ristad *et al.* approach.

This can be computed by means of an auxiliary function α as:

$$\begin{aligned}\alpha(x,y) = {} & [1]_{x=\epsilon\wedge y=\epsilon} \\ & + [c(a,b)\cdot\alpha(x',y')]_{x=x'a\wedge y=y'b} \\ & + [c(a,\epsilon)\cdot\alpha(x,y')]_{x=x'a} \\ & + [c(\epsilon,b)\cdot\alpha(x',y)]_{y=y'b}\end{aligned}$$

And then,

$$p(y|x) = \alpha(x,y)\gamma$$

Of course, in order to have a well defined probability we have to assure the normalization condition:

$$\sum_{y\in Y^*} p(y|x) = 1 \ \forall x \in X^*$$

It can be seen [8] that the following condition over the cost function assures that the stochastic edit distance is well defined:

$$\begin{aligned}\gamma > 0, \quad c(a,b), c(a,\epsilon), c(\epsilon,b) \geq 0 & \qquad \forall a\in X, b\in Y \\ \sum_{b\in Y} c(\epsilon,a) + \sum_{b\in Y} c(a,b) + c(a,\epsilon) = 1 & \qquad \forall a \in X \\ \sum_{b\in Y} c(\epsilon,b) + \gamma = 1 &\end{aligned}$$

Finally, in order to have a dissimilarity measure, the *stochastic edit distance* is defined as:

$$d(x,y) = -\log p(y|x)$$

Oncina and Sebban, in 2006 [8], proposed an expectation-maximization based algorithm [9] to learn the probabilities of the cost function from a training set.

3 Median String and Set Median

Given a set $S \subset M$ and a dissimilarity function $d : M \times M \rightarrow \mathbf{R}$. The (*geometric* or *generalized*) *median* element of the set S is defined as the point in M that minimizes the sum of distances to the elements in S. That is,

$$m = \operatorname*{argmin}_{y\in M} \sum_{x\in S} d(y,x) \tag{2}$$

When the set M is an Euclidean space there exists fast iterative algorithms, like the Weiszfeld's algorithm, to find it.

Given an alphabet X, when the set M is X^* and the dissimilarity function is the edit distance it has been show that the problem becomes NP-Hard[2]. In such cases it is costumary to restrict ourself to search the median in the set S, the obtained element is then called the set-median.

Unfortunately, Martínez-Hinarejos *et al.* [5] [10] showed that the median string is better representative of a given set than the set median. In their work [11], they proposed several approximation methods to find the median. These methods are based on an iterative process of perturbation (see [4]) over an initial string. The most successful one, the joined iterative approach, applies each possible edit operation to each position of a string u (initially ϵ). From these strings, the one that minimizes the sum of the distances to the elements of S is selected, for the next iteration. The process is repeated until no changes are obtained.

In our case we are interested in the stochastic distance. It is not known if, using this distance, the problem of finding the median string of a set is also a NP-Hard problem.

In the next section an Expectation Maximization based method to found the median is proposed. In the experiments section it is compared with a straight forward adaptation to stochastic distances of the Martínez-Hinarejos *et al.* technique.

4 Stochastic Approach to Median String

Given an alphabet X of size n we are going to represent a multisymbol $\mathbf{a} = (\mathbf{a}_1, \ldots, \mathbf{a}_n)$ as a stochastic vector in $\mathbf{R}^n$, that is, $0 \leq \mathbf{a}_i \leq 1$ and $\sum_{i=1}^n \mathbf{a}_i = 1$. A multisymbol associates a probability to any symbol of X. Let $X = \{a_1, \ldots, a_n\}$ and a multisymbol $\mathbf{a}$ we say that $\mathbf{a}_i$ is the probability associated to the symbol $a_i \in X$ ($p_{\mathbf{a}}(a_i) = \mathbf{a}_{a_i}$).

Let $M_X = \{(\mathbf{a}_1, \ldots, \mathbf{a}_n) : |X| = n, 0 \leq \mathbf{a}_i \leq 1, \sum_{i=1}^n \mathbf{a}_i = 1\}$ be a multisymbol alphabet. In the same way as in the case of symbol strings, a multisymbol string is any finite concatenation of elements in M_X. We are going to represent by $\mathbf{x}_i$ the i-th multisymbol in the string $\mathbf{x}$ and by $\mathbf{x}_{i,j}$ the j-th component of the multisymbol $\mathbf{x}_i$. On the following we are going to use $\mathbf{a}, \mathbf{b}, \ldots$ to denote multisymbols, and $\mathbf{x}, \mathbf{y}, \ldots$ to denote multisymbol strings.

Note that a multistring $\mathbf{x}$ of length n ($|\mathbf{x}| = n$) defines a distribution probability $p_{\mathbf{x}}$ over X^n (where $\forall x \in X^n$, $p(x_i) = \mathbf{x}_{i,x_i}$). That is, $p_{\mathbf{x}}(x) \geq 0 \; \forall x \in X^n$ and $\sum_{x \in X^n} p_{\mathbf{x}}(x) = 1$.

Like in the plain string case, The (conditional) stochastic edit distance is given by a 4-tuple (X, Y, c, γ) where X and Y are the input and output alphabet respectively, $c : E \to \mathbf{R}$ is the cost function and $\gamma \in \mathbf{R}$.

The probability of generating the multistring $\mathbf{y}$ from the multistring $\mathbf{x}$ is defined as:

$$p(\mathbf{y}|\mathbf{x}) = \sum_{y \in \Sigma^{|\mathbf{y}|}} \sum_{x \in \Sigma^{|\mathbf{x}|}} p_{\mathbf{y}}(y) p_{\mathbf{x}}(x) p(y|x)$$

Note that the distance in plain string is a special case of this one when all the probabilities of each multisymbol are zero except the corresponding to the actual symbol in the string.

Similarly to the string case, this probability can be computed recursively as follows:

$$p(\mathbf{y}|\mathbf{x}) = \alpha(\mathbf{x},\mathbf{y})\gamma$$

where

$$\begin{aligned}\alpha(\mathbf{x},\mathbf{y}) = {} & [1]_{\mathbf{x}=\epsilon,\mathbf{y}=\epsilon} \\ & + [c(\mathbf{a},\mathbf{b}) \cdot \alpha(\mathbf{x}',\mathbf{y}')]_{\mathbf{x}=\mathbf{x}'\mathbf{a}\wedge\mathbf{y}=\mathbf{y}'\mathbf{b}} \\ & + [c(\mathbf{a},\epsilon) \cdot \alpha(\mathbf{x}',\mathbf{y})]_{\mathbf{x}=\mathbf{x}'\mathbf{a}} \\ & + [c(\epsilon,\mathbf{b}) \cdot \alpha(\mathbf{x},\mathbf{y}')]_{\mathbf{y}=\mathbf{y}'\mathbf{b}}\end{aligned} \tag{3}$$

and where

$$c(\mathbf{a},\epsilon) = \sum_{i=1}^{n} c(a_i,\epsilon)\mathbf{a}_i \tag{4}$$

$$c(\epsilon,\mathbf{b}) = \sum_{j=1}^{n} c(\epsilon,b_j)\mathbf{b}_j \tag{5}$$

$$c(\mathbf{a},\mathbf{b}) = \sum_{i=1}^{n}\sum_{j=1}^{n} c(a_i,b_j)\mathbf{a}_i\mathbf{b}_j \tag{6}$$

This probability can be computed in $O(|\mathbf{x}||\mathbf{y}|)$ using dynamic programming techniques.

In a symetric way $p(\mathbf{y}|\mathbf{x})$ can be recursively computed by means of an auxiliary function β as:

$$p(\mathbf{y}|\mathbf{x}) = \beta(\mathbf{x},\mathbf{y})\gamma$$

where

$$\begin{aligned}\beta(\mathbf{x},\mathbf{y}) = {} & [1]_{\mathbf{x}=\epsilon,\mathbf{y}=\epsilon} \\ & + [c(\mathbf{a},\mathbf{b}) \cdot \beta(\mathbf{x}',\mathbf{y}')]_{\mathbf{x}=\mathbf{a}\mathbf{x}'\wedge\mathbf{y}=\mathbf{b}\mathbf{y}'} \\ & + [c(\mathbf{a},\epsilon) \cdot \beta(\mathbf{x}',\mathbf{y})]_{\mathbf{x}=\mathbf{a}\mathbf{x}'} \\ & + [c(\epsilon,\mathbf{b}) \cdot \beta(\mathbf{x},\mathbf{y}')]_{\mathbf{y}=\mathbf{b}\mathbf{y}'}\end{aligned} \tag{7}$$

The stochastic edit distance can be extended to multistrings as:

$$d(\mathbf{x},\mathbf{y}) = -\log p(\mathbf{y}|\mathbf{x})$$

Given a set of strings S we are interested in finding the median multistring $\mathbf{x}$, that is:

$$\begin{aligned}\mathbf{x} &= \operatorname*{argmin}_{\mathbf{x}\in M_X} \sum_{y\in S} d(\mathbf{x}, y)\\ &= \operatorname*{argmin}_{\mathbf{x}\in M_X} - \sum_{y\in S} \ln p(y|\mathbf{x})\\ &= \operatorname*{argmax}_{\mathbf{x}\in M_X} \prod_{y\in S} p(y|\mathbf{x})\end{aligned}$$

Then the problem of finding the median multistring can be stated as a maximum likelihood problem.

Now, we are going restrict the problem to search the multistring $\mathbf{x}$ of a fixed length that maximizes the likelihood and then iterate for each possible length. To do that the standard expectation maximization algorithm is used.

It is easy to see in equation 3 that the parameters of $\mathbf{x}$ are only used when deleting (eq. 4) and substituting (eq. 6). Then the expectation $\bar{\mathbf{x}}_{i,j}$ of each parameter of $\mathbf{x}$ can be computed as:

$$\begin{aligned}\bar{\mathbf{x}}_{i,k} = &\sum_{j=1}^{|y|} \frac{\alpha(\mathbf{x}_{1\ldots i}, y_{1\ldots j})\mathbf{x}_{i,k} c(a_k, \epsilon)\beta(\mathbf{x}_{i+1\ldots n}, y_{j\ldots|y|})\gamma}{p(y|\mathbf{x})}\\ &+ \sum_{j=1}^{|y|} \frac{\alpha(\mathbf{x}_{1\ldots i}, y_{1\ldots j})\mathbf{x}_{i,k} c(a_k, y_j)\beta(\mathbf{x}_{i+1\ldots n}, y_{j+1\ldots|y|})\gamma}{p(y|\mathbf{x})}\end{aligned}$$

And the maximization consists in renormalizing. That is,

$$\mathbf{x}_{i,j} = \frac{\bar{\mathbf{x}}_{i,j}}{\sum_{k=1}^{|X|} \bar{\mathbf{x}}_{i,k}}$$

As usual, both steps are repeated until a convergence criterion is reached.

5 Experiments and Results

5.1 *Copenhagen* Corpus

The database used to develop the experiments is the *Copenhagen chromosome dataset*. Each chromosome is depicted by a digitized image which was automatically transformed into a string through the procedure illustrated in figure 1. This procedure begins with the transformation of the images into its idealized profiles. Then, each profile is mapped into strings over the alphabet {1,2,3,4,5,6}. After that, these strings are coded in order to represent signed differences of successive symbols over the alphabet Σ = {e,d,c,b,a,=,A,B,C,D,E}. Taking into account that "a" correspond to difference of -1, "A" of +1, "=" of 0, and so on. For a complete reference to this procedure see [12].

The dataset has 200 strings per class and there are 22 non-sex chromosome types, so the dataset is formed by 4400 samples. Moreover, the behavior of the

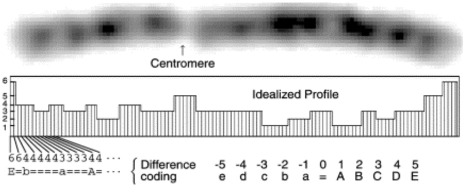

Fig. 1. Image of the chromosomes preprocessing

algorithms is evaluated based on a two-fold cross-validation. Accordingly, each class is divided into two equal-size sets with 100 samples each, then the F_j folds are formed by 2200 samples, where $j \in \{1, 2\}$.

5.2 Experiment 0: Learning the Stochastic Edit Distance

In order to use the stochastic edit distance we need to fix the edit operation probabilities. A similar approach to the one used in [8] was followed. Let P_j be a set of $(x, \mathrm{NN}(x))$ chromosomes pairs. Where each pair into the P_j set is formed by a sample x from F_l and its nearest neighbor $\mathrm{NN}(x)$. The P_j set was used as training set to learn Stochastic Distance. A different stochastic edit distance was learned for each set P_j.

5.3 Experiment 1: Clustering with *k*-Medians

Following the guidelines made in Martínez-Hinarejos *et al.* work we begin by making some experiments to know how much better are our medians with respect to the set-median and the medians found by the algorithm of Martinez-Hinarejos *et al.*. To do that the *k-medians* clustering algorithm along with the *minmax* initialization was use [13].

In the experiments, the median computation in k-medians was replaced by each of the three median computation algorithms that we are comparing. The number of clusters was increased from 1 to 100. As a quality measure the sum of the distances from medians to the elements in its cluster was used.

It can be observed in figure 2 that the worst results are obtained when the set-median is used. Of course, the sum of distances for the three algorithms converges as the number of prototypes grows.

5.4 Experiment 2: Classification with *k*-Nearest Neighbor

The main goal of this experiment is the evaluation of the k-Nearest Neighbor (k-NN) classification algorithm. According to this, the three versions of the c-medians algorithms were applied to each class to obtain c representative of the class (c varying from 1 to 100). Then the k-NN classification algorithm (for $k \in \{1, 3, 6\}$) was applied to classify the prototypes of an independent fold.

The classification error rates for each algorithm plus the k-NN using the whole set of prototypes are shown in figure 3.

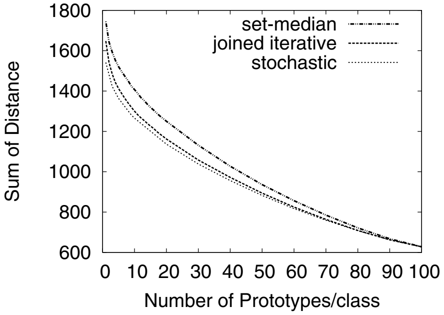

Fig. 2. Minimum sum of distances from the prototypes to its represented class

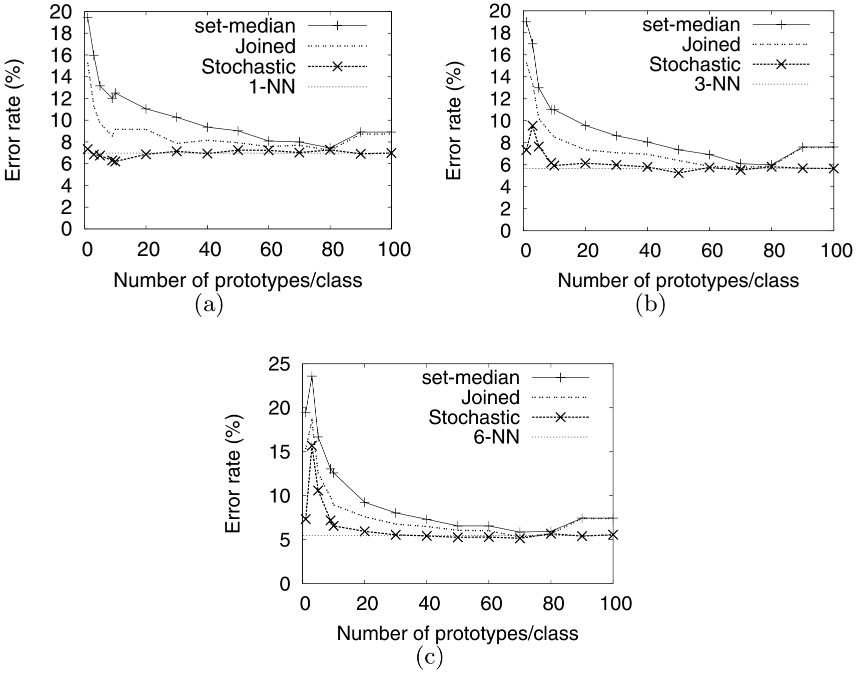

Fig. 3. Error rate of classification using 1-NN (a), 3-NN (b) and 6-NN (c)

Of course, using the whole set of prototypes obtains the lowest error rates, but, among the algorithms that are performing a prototype selection the one based in multistring obtains the lowest error rates.

6 Conclusions

In this work we have used the concept of a multistring in order to compute the median of a set of strings when using the stochastic edit distance. We have show that the problem can be stated as a maximum likelihood search and then, we used the Expectation Maximization algorithm.

In the experimental section we have shown that this approach is quite effective when comparing to it the set median or the generalized median.

Acknowledgments. The authors thank the Spanish CICyT for partial support of this work through projects DPI2006-15542-C04-01, the IST Program of the European Community, under the PASCAL Network of Excellence, IST–2002-506778, and the program CONSOLIDER INGENIO 2010 (CSD2007-00018).

References

1. Levenshtein, V.I.: Binary codes capable of correcting deletions, insertions and reversals. Soviet Physics Doklady 10, 707–710 (1966)
2. de la Higuera, C., Casacuberta, F.: Topology of strings: Median string is NP-complete. Theoretical Computer Science 230, 39–48 (2000)
3. Fu, K.S.: Syntactical Pattern Recognition and Applications. Prentice-Hall, Englewood Cliffs (1982)
4. Kohonen, T.: Median Strings. PRL 3, 309–313 (1985)
5. Martínez-Hinarejos, C.D., Juan, A., Casacuberta, F.: Use of Median String for Classification. In: ICPR, pp. 2903–2906 (2000)
6. Martínez-Hinarejos, C.D., Juan, A., Casacuberta, F.: Median Strings for k-nearest neighbour classification. Pattern Recog. Lett. 24, 173–181 (2003)
7. Ristad, E.S., Yianilos, P.N.: Learning String-Edit Distance. IEEE Trans. Pattern Anal. Mach. Intell. 20, 522–532 (1998)
8. Oncina, J., Sebban, M.: Learning stochastic edit distance: Application in handwritten character recognition. Pattern Recognition 39, 1575–1587 (2006)
9. Dempster, A.P., Laird, N.M., Rubin, D.B.: Maximum Likelihood from Incomplete Data via the EM Algorithm. Journal of the Royal Statistical Society. Series B (Methodological) 39, 1–38 (1977)
10. Martínez-Hinarejos, C.D., Juan, A., Casacuberta, F., Mollineda, R.A.: Reducing the Computational Cost of Computing Approximated Median Strings. In: IAPR International Workshop on Structural, Syntactic, and Statistical Pattern Recognition, pp. 47–55. Springer, London (2002)
11. Martínez-Hinarejos, C.D.: La cadena media y su aplicación en reconocimiento de formas. Phd. Thesis. Departamento de Sistemas Informáticos y Computación, Universidad Politécnica de Valencia (2003)
12. Granum, E., Thomason, M.G.: Automatically inferred markov network models for classification of chromosomal band pattern structures. Cytometry 11, 26–39 (1990)
13. Juan, A., Vidal, E.: Comparison of Four Initialization Techniques for the K - Medians Clustering Algorithm. In: Joint IAPR International Workshops on Advances in Pattern Recognition, pp. 842–852. Springer, London (2000)