Pattern Analysis and Applications
https://doi.org/10.1007/s10044-018-0734-z

THEORETICAL ADVANCES

An accurate HMM-based similarity measure between finite sets of histograms

Sylvain Iloga[1,2,3] · **Olivier Romain**[2] · **Maurice Tchuenté**[3]

Received: 19 October 2017 / Accepted: 19 July 2018

Abstract
Histogram analysis has nowadays gain in interest, and a lot of work yet address this task. In most of the existing approaches, histograms are manipulated as simple vectors or as statistic distributions. As a consequence, only the bin values of the histograms are mostly considered and the histograms visual shapes are generally neglected. In this paper, hidden Markov models (HMMs) are associated with finite sets of histograms to capture both: the bin values and the visual shapes of the histograms contained in these sets, regardless of their bin sizes. The similarity rate between these HMMs is then used to compare two finite sets of histograms. Experimented in several areas within and beyond machine learning, the proposed approach exhibited relevant performances which outperformed the existing work in the hierarchical classification of the databases *GTZAN+* and *Corel*.

Keywords Histogram comparison · Hidden Markov models · Color image comparison · Comparison of function curves · Automatic taxonomy generation · Hierarchical classification · Text document comparison

1 Introduction

Histogram-based local descriptors are nowadays used in several domains. In music processing, they are used to compute music descriptors like rhythm histograms [1]. They are also thoroughly used in image processing with image descriptors like color histograms [2], edge direction histograms [3] or Tamura coarseness histograms [4]. Therefore, histogram analysis has become an unavoidable exercise on which a lot of works have already been devoted. Histogram analysis can be divided into two main axes: (1) histogram comparison where several distances and similarities between two histograms are proposed [2, 5–15]; (2) histogram modeling where models are designed to capture some histogram properties [16, 17]. These models are generally used for categorization purposes.

One of the highest difficulties in comparing two histograms is the choice of the suitable distance or similarity measure. This is due to the great number of existing measures. This difficulty is accentuated by the fact that every existing measure captures a specific similarity property between the two histograms h_1 and h_2 that must be compared. As an example, the *Euclidean distance* measures the straight-line distance between h_1 and h_2, while the *Bhattacharyya* distance [18] rather approximates the amount of overlap between them. Furthermore, most of the existing measures only enable to compare histograms with identical number of bins, whereas comparing histograms with different bin sizes may be very useful. The *Earth Movers* distance [13] addresses this situation.

In most of the existing approaches related to histogram comparison, a histogram h composed of n bins is manipulated as a pure n-dimensional vector of $\mathbb{R}^n$. Therefore, the distance measures that are commonly used to compare vectors in $\mathbb{R}^n$ are also used to compare histograms. Sometimes

✉ Sylvain Iloga
sylvain.iloga@yahoo.fr

Olivier Romain
olivier.romain@gmail.com

Maurice Tchuenté
maurice.tchuente@gmail.com

[1] Higher Teachers' Training College, Department of Computer Science, University of Maroua, P.O.box 55, Maroua, Cameroon

[2] UMR 8051, ETIS Laboratory, CNRS, ENSEA, University of Cergy-Pontoise, 6 avenue du Ponceau, 95000 Cergy, France

[3] IRD UMI 209, UMMISCO-LIRIMA Laboratory, University of Yaoundé 1, P.O.box 337, Yaounde, Cameroon

Published online: 30 July 2018

histograms are rather manipulated as statistic distributions and are compared with distance measures dedicated to statistic distributions like the χ^2-*statistics* distance. The use of such distance measures leads to a huge lost of information, because a histogram is basically something more subtle. Indeed, it can be seen as an ordered sequence of bin values whose variations are meaningful. These variations determine the visual shape of the histogram. Unlike vectors or statistic distributions whose components can be compared in any order, histograms embed a natural sequentiality in the occurrences of their bin values that must be considered to measure their similarity.

This observation is confirmed when the graphical representations of vectors and histograms are considered. A *n*-dimensional vector is generally represented by an arrow in $\mathbb{R}^n$. But a histogram is represented by a bar diagram where: (1) the value of the *k*th bin determines the height of the *k*th bar and (2) the sequential order in which the bin values vary determines the shape of the histogram. Consider, for example, the three following histograms $h_1 = [5, 4, 2]$, $h_2 = [2, 4, 5]$, $h_3 = [4, 8, 3]$, where h_2 is h_1 taken in reverse order and h_3 is completely different from the two other histograms. The *Euclidean* distance of the pairs (h_1, h_2) and (h_1, h_3) is $3\sqrt{2}$, which means that h_1 is equidistant to the two other histograms. But this information is not meaningful regarding the graphical representations of h_1, h_2 and h_3 in Fig. 1a–c.

The value $3\sqrt{2}$ is the distance between these histograms according to their bin values, but it includes no information about their visual shapes. Indeed, it is not easy even for a human being, to compare the shapes of these histograms because: h_1 always decreases, h_2 always increases, but h_3 first increases and then decreases.

Histogram modeling often deals with classification problems where many classes are available, each class c_i being represented by $|c_i|$ histograms (instances) with identical number of bins. A model is generally initialized and trained for each class c_i to capture the inner properties of its instances. An obvious drawback of these techniques is the elevated time cost of the model training. Some authors have studied models without long training phases. This is the case in [17] where discrete Markov models (DMMs) are used to model

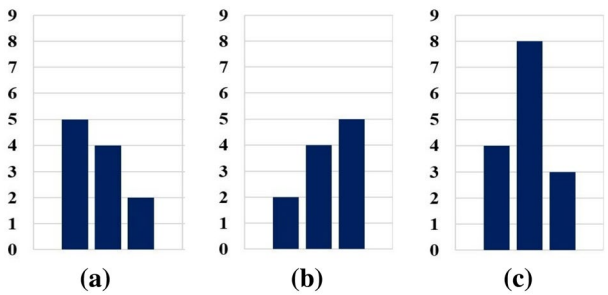

Fig. 1 Example of histograms to be compared. **a** h_1. **b** h_2. **c** h_3

histograms in image categorization. The color histogram of each image of a class c_i is first transformed into a *quasi-histogram*, and then, the quasi-histograms of all the images in c_i are used to estimate the two parameters of the model associated with c_i. To determine the class of an image I, its quasi-histogram q_I is first computed and then I is associated with the class whose model produced the highest probability to observe q_I. The author also proposed the possibility to combine these DMMs with region hidden Markov models (HMMs) to slightly improve the performances that method.

Another limitation of existing histogram modeling approaches is the fact that in some cases, the training of the model of a class c_i depends on the data of another class c_j. This limitation can be observed in [16] where sequences of histograms are modeled by *sequential patterns* to perform hierarchical music genre recognition. Such training dependencies imply that the resulting model of c_i does not exclusively capture the properties of its instances. This is not correct because the existence of class c_i does not depend on the existence of another class.

Beyond all these limitations, we have found no approach specially designed to compare two finite sets of histograms. Indeed, existing approaches related to histogram comparison can only compare two single histograms with identical bin size. It is true that the problem of comparing two finite sets H_1 and H_2 containing histograms with identical bin size can be basically tackled by considering each set as a cluster. The comparison can then be realized by applying common cluster distances like the *Minimum* or the *Maximum* histogram-based distance between all the pairs (h_1, h_2) with $h_1 \in H_1$ and $h_2 \in H_2$. But such an approach is limited because the final result does take into account the specific properties of all the histograms in H_1 and H_2.

This paper addresses histogram modeling and histogram comparison with the advantage that the proposed approach attempts to avoid most of the aforementioned limitations. More precisely, HMMs are trained to capture: (1) the bin values of the histograms and (2) the sequential bin values variations of the histograms in order to take their shapes into account. Since HMMs can be trained either for one single sequence or for multiple sequences, the proposed approach produces a robust model for one single histogram as well as for a finite set of histograms. The similarity between these HMMs is later computed to perform histogram comparison. Given that the training of a HMM can be performed whatever are the lengths of the training sequences, the proposed approach can consequently realize the comparison regardless of the bin sizes of the histograms. The design of each HMM only involves the data of the concerned histograms. Therefore, there are no training dependencies. The performances of the proposed approach are finally evaluated in color image comparison, in the comparison of function curves, in text document comparison and in automatic taxonomy generation.

The rest of this paper is organized as follows: Sect. 2 presents the sate of the art on histogram comparison, followed by a detailed presentation of the proposed approach in Sect. 3. Experimental results are presented in Sect. 4, and the last section is devoted to the conclusion.

2 State of the art

2.1 Related work

Four main categories of similarity and distance measures between two histograms can be distinguished in the literature [18]. The first three categories apply bin-to-bin functions, and the last category is empowered by the use of cross-bin information. Table 1 lists up to 20 existing similarity and distance measures between two histograms, each measure being associated with its corresponding category. Beside the content of Table 1, numerous other similarity and distance measures between two histograms have yet been proposed. As an example, the authors of [20] proposed a distance between sets of measurement values as an interesting measure of dissimilarity between two histograms. A fast algorithm based on the concept of *histograms signatures* was also proposed in [21] to compute the distance between histograms. In their book published in 2016, Ionescu and Popescu [22] surveyed many other existing distance measures between histograms.

Some other researchers rather focused on distance metric learning approaches which greatly improve the performances of metrics-dependent algorithms such as the *k-means* clustering or the *K-NN* algorithms. These approaches have therefore gained popularity in many areas within machine learning such as in pattern recognition and image analysis [23,

Table 1 Some relevant existing similarity and distance measures between two histograms h_1 and h_2 grouped in four categories

Category	Name	Acro.	Short description
Derived from heuristics	Manhattan	L_1	Sum of bin-to-bin variations between h_1 and h_2
	Euclidean	L_2	Straight-line distance between h_1 and h_2
	Tchebychev	L_∞	Maximum bin-to-bin variation between h_1 and h_2
	Minkwoski	L_p	Generalization of L_1, L_2 and L_∞
	Intersection [2]	$D_\cap$	Sum of minimum bin-to-bin values between h_1 and h_2
	Cosine	*CO*	The cosine of the angle between h_1 and h_2 extracted from their dot product
	Pearson's correlation [5]	*CR*	Linear dependance between h_1 and h_2
	Canberra [6]	*CB*	Weighted version of the L_1 distance
Derived from nonparametric test statistics	Hellinger [19]	*HL*	Quantifies the similarity between h_1 and h_2
	Kolmogorov–Smirnov [18]	*KS*	Maximal divergence between two cumulative distributions
	Match [18]	*MA*	Sum of absolute distances between two cumulative distributions
	Cramer–Von Mises [18]	*CM*	Penalizes the divergence of between two cumulative distributions quadratically as it sums them
	χ^2-statistics [8]	*CS*	Evaluates how likely it is that any observed difference between two distributions arose by chance
	Bhattacharyya [18]	*BH*	Approximates the amount of overlap between two distributions
Derived from information-theoretic divergence	Kullback–Leibler divergence [9]	*KL*	Measures how inefficient on average it would be to code h_1 using h_2 as true distribution for coding
	Jeffrey divergence [18]	*JD*	Stable version of the *KL* divergence
Using cross-bin information	Quadratic form [10]	*QF*	The similarity match between h_1 and h_2
	Quadratic-Chi [11]	*QC*	Utilizes the normalization power of the Chi-square along with cross-bin relationship presented by the *QF* distance
	Diffusion [12]	*DF*	Models the distance between two histograms as a temperature field and considers the diffusion process on the field
	Earth Movers [13]	*EM*	Least amount of work needed to transport earth or mass from one distribution to the other

The acronym used in the literature for each similarity/distance is in the third column

24]. In [15], a nonlinear distance metric learning approach dedicated to histograms is proposed. The authors of that paper generalized the bin-to-bin χ^2-*statistics* distance in order to learn a metric that strictly preserves the histograms properties. This approach exhibited more reliable results than the χ^2-*statistics* distance.

Unfortunately, all the aforementioned approaches can only compare two single histograms with identical bin size. Indeed, despite our research we did not find an existing approach related to histogram comparison capable of comparing two finite sets containing histograms with different bin sizes. However, there are many problems for which such a capability is necessary. Suppose, for example, that we want to compare the efficiency of two basketball players x and y during their respective *entire* careers in professional league. To achieve this objective, we decide to construct two histograms h_x and h_y such that the ith bin of each histogram represents the number of points that the considered player scored during his ith match in the professional league. The experience starts from the first match and ends with the last match of each player in the professional league. Since it is very unlikely that both players count the same number of matches at the end of their respective professional careers, this experience leads to the comparison of two histograms h_x and h_y with different numbers of bins. Furthermore, if the experience no longer targets only two different players x and y, but rather targets two sets $X = \{x_1, \ldots, x_m\}$ and $Y = \{y_1, \ldots, y_n\}$, respectively, composed of m and n players, then it will now be a question of comparing two finite sets $H_X = \{h_{x_1}, \ldots, h_{x_m}\}$ and $H_Y = \{h_{y_1}, \ldots, h_{y_n}\}$ containing histograms with different numbers of bins. In such a context, all the related works cited in this paper are not applicable. This justifies the interest of the proposed approach.

2.2 Hidden Markov models

2.2.1 Definition of a HMM

A HMM $\lambda = \{A, B, \pi\}$ is characterized by:

1. Its number N of states. The set of states is noted $S = \{S_1, S_2, \ldots, S_N\}$, and generally, the state of the model at time t is noted $q_t \in S$.
2. Its number M of observation symbols. The set of symbols is noted $V = \{v_1, v_2, \ldots, v_M\}$, and the symbol observed at time t is generally noted $O_t \in V$.
3. Its state transition probability distribution $A = \{a_{ij}\}$ where $a_{ij} = P(q_{t+1} = S_j | q_t = S_i)$ with $1 \leq i, j \leq N$.
4. Its observation symbols probabilities distributions $B = \{b_i(k)\}$ in each state S_i where $b_i(k) = P(v_k \text{ at time } t | q_t = S_i)$ with $1 \leq i \leq N$ and $1 \leq k \leq M$.
5. Its initial state probability distribution $\pi = \{\pi_i\}$ where $\pi_i = P(q_1 = S_i)$ with $1 \leq i \leq N$.

2.2.2 Sequence generation with a HMM

Equation 1 shows how a HMM $\lambda = \{A, B, \pi\}$ can generate a sequence $O = O_1, O_2, \ldots, O_T$ composed of T observation symbols. In the rest of this paper, such a representation will be called a *generated Markov chain*.

$$\begin{array}{lccccccc} \text{(symbols)} & O_1 & & O_2 & & \ldots & & O_T \\ & \uparrow & & \uparrow & & \vdots & & \uparrow \\ \text{(states)} & q_1 & \rightarrow & q_2 & \rightarrow & \ldots & \rightarrow & q_T \end{array} \quad (1)$$

This generated Markov chain is obtained using the following algorithm:

1. Select an initial state $S_j \in S$ with respect to the distribution π and set $t = 0$.
2. Set $t = t + 1$, then edit the current state to $q_t = S_j$
3. Select a symbol $O_t \in V$ observed at state q_t with respect to the distributions in B.
4. If $(t < T)$ **go to** step 5, else **terminate**.
5. Select a state transition to be realized from state q_t to another state $S_j \in S$ with respect to the distribution A, then **go to** step 2.

2.2.3 HMM probability computing and training

Consider an observation sequence $O = O_1, O_2, \ldots, O_T$ and a HMM $\lambda = \{A, B, \pi\}$. The probability $P(O|\lambda)$ to observe O given λ is efficiently calculated by the *Forward–Backward* algorithm [25]. This algorithm runs in $\theta(T.N^2)$.

Given an observation sequence $O = O_1, O_2, \ldots, O_T$, the parameters of a HMM $\lambda = \{A, B, \pi\}$ can be re-estimated in order to maximize the value of $P(O|\bar{\lambda})$, where $\bar{\lambda} = \{\bar{A}, \bar{B}, \bar{\pi}\}$ is the re-estimated model. This re-estimation is done by the *Baum–Welch* algorithm [25]. This algorithm improves the probability of O being observed from the model by iteratively using $\bar{\lambda}$ in place of λ and repeating this re-estimation until $\bar{\lambda} = \lambda$, or until a user-defined maximum number of iterations is reached.

It is also possible to train a HMM to maximize the value of $P(O|\bar{\lambda}) = \sum_{k=1}^{K} P(O^{(k)}|\bar{\lambda})$ where $O = \{O^{(1)}, \ldots, O^{(K)}\}$ is a set of K observation sequences and $O^{(k)} = O_1^{(k)} \ldots O_{T_k}^{(k)}$ is the kth observation sequence of O. In the case of multiple sequences, only the distributions A and B are re-estimated by the *Baum–Welch* algorithm because $\bar{\pi}$ can be statistically determined from the initial states of the K observed sequences.

2.2.4 Stationary distribution of a HMM

A vector $\varphi = (\varphi_1, \ldots, \varphi_N)$ is a stationary distribution of a HMM $\lambda = \{A, B, \pi\}$ if: (1) $\sum_j \varphi_j = 1$, (2) $\forall j, \varphi_j \geq 0$, (3)

$\varphi = \varphi.A \Leftrightarrow \left[\varphi_j = \sum_i (\varphi_i.a_{ij}), \forall j\right]$. φ_j estimates of the overall proportion of time spent by λ in state j. This distribution can be extracted from any line of the matrix A^k when $k \to +\infty$.

2.2.5 Similarity between two HMMs

Numerous distance and similarity measures between two HMMs have yet been proposed in existing work [26–32]. All these metrics are limited and their limitations are reviewed in [33] where an accurate low-complexity similarity measure between two HMMs λ and λ' was proposed. This measure evaluates the probability that λ and λ' will generate identical observation sequences following the algorithm described in Sect. 2.2.2. The authors of [33] proved that this measure requires only a small fraction of the computation time of existing measures. For all these reasons, it is this measure that has been selected to compare HMMs in this paper. Let us note $Sim(\lambda, \lambda') \in [0, 1]$ the selected similarity measure between two HMMs $\lambda = \{A, B, \pi\}$ and $\lambda' = \{A', B', \pi'\}$ proposed in [33]. If n and n' are, respectively, the number of states of λ and λ', then Eq. 2 shows how to compute $\text{Sim}(\lambda, \lambda')$.

$$\text{Sim}(\lambda, \lambda') = \frac{1}{2}\left[\frac{1}{n}\sum_{j=1}^{n} G(\text{row}_j) + \frac{1}{n'}\sum_{k=1}^{n'} G(\text{col}_k)\right]. \tag{2}$$

In Eq. 2:

1. row_j and col_k are, respectively, the jth row and the kth column of the state correspondence matrix $Q = \{q_{ii'}\}$ between λ and λ'. For every state i of λ and i' of λ', $q_{ii'}$ is calculated using Eq. 3 where φ and φ' are the stationary distributions of λ and λ'. The value $S(i, i')$ is calculated with Eq. 4. In this work, the parameters $k = 2$ and $\alpha = 0.5$ have been used to compute $S(i, i')$.

$$q_{ii'} = \frac{\varphi_i \varphi'_{i'} S(i, i')}{\sum_{\forall i,i'} \varphi_i \varphi'_{i'} S(i, i')}, \tag{3}$$

$$\begin{aligned} &S(i, i') = e^{-k.D_\alpha(b_i, b'_{i'})}, \text{ where} \\ &D_\alpha(b_i, b'_{i'}) = \frac{1}{\alpha - 1} \log\left(\sum_k (b_i(k))^\alpha . (b'_{i'}(k))^{1-\alpha}\right). \end{aligned} \tag{4}$$

2. The function $G(x)$ is the normalized *Gini Index* defined in Eq. 5. In that equation, m is the number of components x, $||x||_1$ is the sum of the components of x and $x_{(k)}$ is the kth smallest element of x such that $x_{(1)} \leq x_{(2)} \leq \cdots \leq x_{(m)}$. If x is the null vector, then $G(x) = 0$.

$$G(x) = \frac{m}{m-1} - 2\sum_{k=1}^{m} \frac{x_{(k)}}{||x||_1}\left(\frac{m - k + \frac{1}{2}}{m-1}\right). \tag{5}$$

2.3 Problem statement

As given in Sect. 2.1, there exist a huge number of distance and similarity measures between histograms. The problem is that in all the existing approaches we reviewed, only two single histograms with identical bin sizes are compared. Furthermore, existing histograms modeling approaches suffer from training dependencies. The goal of this paper is to propose a HMM-based approach that can accurately model finite sets of histograms without training dependencies. The similarity rate between these HMMs is later computed to perform histogram comparison. The proposed histogram modeling approach takes into account the bin values as well as the visual shapes of the histograms, whatever are their bin sizes.

The choice of HMMs in this paper is first motivated by the fact that they embed a natural temporality; thus, they are adequate to capture the sequential variations of bin values in the histograms. Additionally, HMMs are managed by robust algorithms whose efficiency and accuracy do not more need to be demonstrated.

3 The proposed approach

3.1 Main idea of histograms modeling

Equation 1 depicts a generated Markov chain produced by a HMM λ. When a deep attention is paid to the appearance of this generated Markov chain, one can observe a high similarity degree with the visual shape of a histogram. Indeed, if the symbols O_t $(1 \leq t \leq T)$ are positive real numbers and if the height of each up arrow is proportional to the value of the symbol on which it points, then Eq. 1 becomes a histogram with the states q_t as bin labels.

It is from this observation that arises the idea to transform each histogram h of a set H into a generated Markov chain similar to the one depicted in Eq. 1. The resulting generated Markov chains will later be used to initialize and train a HMM associated with H. In this work, instead of analyzing each histogram $h \in H$ itself, we rather prefer to analyze its corresponding normalized histogram noted here as $\widehat{h}$. This normalization is crucial because it enables to compare histograms according to their visual shapes, with their bin values always in the range [0, 100] regardless of their real amplitudes. These amplitudes will later be taken into account at the end of the comparing process.

More formally, the normalized histograms associated with the histograms in the set $H = \{h_1, \ldots, h_M\}$ are first calculated and saved in the set $\widehat{H} = \{\widehat{h}_1, \ldots, \widehat{h}_M\}$. Then the generated Markov chains of these normalized histograms are computed and saved in the set $\Delta_{\widehat{H}} = \{\delta_{\widehat{h}_1}, \ldots, \delta_{\widehat{h}_M}\}$. Finally, the underlying HMM $\lambda_{\widehat{H}}$ associated with $\widehat{H}$ is initialized and trained using the *Baum–Welch* algorithm according to the content of $\Delta_{\widehat{H}}$. Figure 2 summarizes the process of HMM design.

3.2 Methodology for histogram comparison

Let H_1 and H_2 be two sets of histograms. In this paper, the similarity between these two sets is evaluated according to the methodology presented in Fig. 3. Two HMMs $\lambda_{\widehat{H}_1}$ and $\lambda_{\widehat{H}_2}$ are first designed to capture the bin values and the visual shapes of their corresponding sets of normalized histograms. Then, the similarity $\widehat{\sigma}(H_1, H_2)$ between these two HMMs is computed and weighted by a suitable amplitude coefficient $\theta(H_1, H_2)$ to obtain the desired similarity rate $\sigma(H_1, H_2)$.

3.3 Histogram normalization

Consider a set $H = \{h_1, \ldots, h_M\}$ of histograms. In the first step of the proposed methodology, each histogram h_i is normalized with respect to the highest bin value of the histograms in H. Equation 6 shows how to compute the normalized histogram $\widehat{h}_i$ corresponding to h_i. When this equation is applied to all the content of H, the set $\widehat{H} = \{\widehat{h}_1, \ldots, \widehat{h}_M\}$ composed of normalized histograms is obtained.

$$\begin{cases} \widehat{h}_i(j) = \frac{100}{H_{\max}} \times h_i(j), \quad 1 \le j \le |h_i| \text{ where} \\ H_{\max} = \max_{1 \le i \le M} \left\{ \max_{1 \le j \le |h_i|} \{h_i(j)\} \right\} \end{cases}. \tag{6}$$

3.4 Histogram transformation

3.4.1 Normalized histogram redefinition

After the normalization step, each normalized histogram $\widehat{h}_i \in \widehat{H}$ with $(1 \le i \le M)$ must be transformed into a generated Markov chain produced by a particular HMM associated with $\widehat{H}$, the parameters of this HMM will later be estimated. During this transformation, the bin values of $\widehat{h}_i$ will be captured as symbols and their variations will be captured as hidden states. The main obstacle to the realization of this transformation is related to the fact that the number of symbols and the number of hidden states of a HMM must always be finite. But this is not actually possible because the bin values in $\widehat{h}_i$ can take any value in the continuous interval [0, 100]. To overcome this difficulty, this interval is split into $(N + 1)$ slices $\{v_0, v_1, \ldots, v_N\}$ as shown in Eq. 7, where N is a user-defined integer.

$$\begin{cases} v_0 = \{0\} \\ v_k =]\frac{100}{N} \times (k-1), \frac{100}{N} \times k], 1 \le k \le N. \end{cases} \tag{7}$$

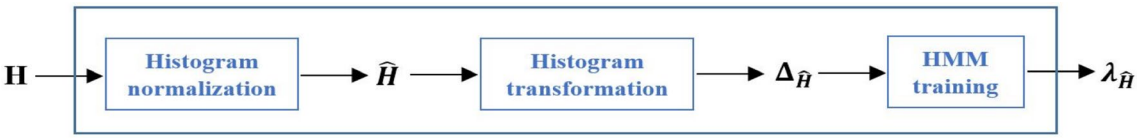

Fig. 2 The HMM design process

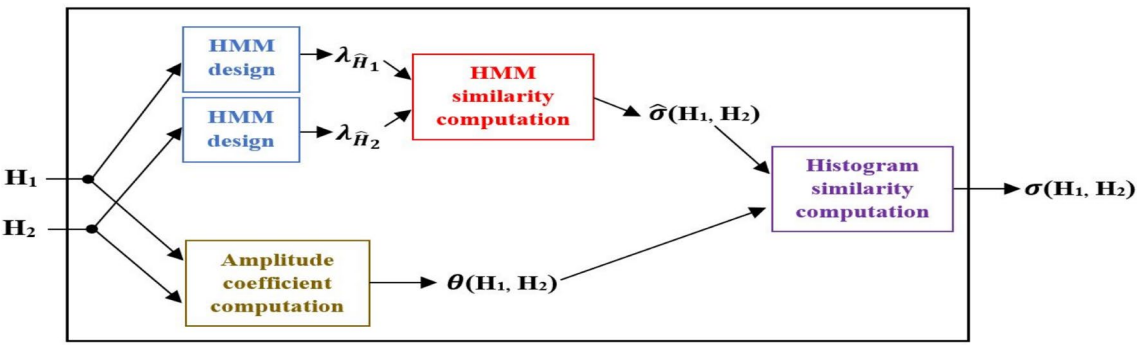

Fig. 3 Proposed methodology to compute the similarity between two finite sets H_1 and H_2 of histograms

If the value of N used to split the range [0, 100] is very high, then the length of each interval v_k becomes tiny and all the elements in v_k become very near to one unique value which is $\frac{100}{N} \times k$. The elements of v_k can therefore be considered as one single element that we identify here by the index k of the interval v_k. This reasoning allows us to define the new histogram noted $\tilde{h}_i$ which is obtained by replacing each bin value of $\widehat{h}_i$ by the index k of the interval v_k to which it belong as shown in Eq. 8. An additional bin whose value is zero is inserted at the index zero of $\tilde{h}_i$, and this insertion enables the computation of an initial bin value variation from $\tilde{h}_i(0)$ to $\tilde{h}_i(1)$. When this principle is applied to all the content of $\widehat{H}$, the set $\tilde{H} = \{\tilde{h_1}, \ldots, \tilde{h_M}\}$ is obtained.

$$\begin{cases} \tilde{h}_i(0) = 0 \\ \left(\tilde{h}_i(j) = k\right) \Leftrightarrow \left(\widehat{h}_i(j) \in v_k\right), \quad 1 \leq j \leq |\widehat{h}_i|. \end{cases} \tag{8}$$

3.4.2 The transformation step

Once $\tilde{H}$ is calculated, the generated Markov chain $\delta_{\widehat{h}_i}$ associated with each histogram $\widehat{h}_i \in \widehat{H}$ is derived according to the principle presented in Fig. 4 where $T_i = |\tilde{h}_i|$. For each histogram $\tilde{h}_i \in \tilde{H}$, this transformation can be summarized as follows:

1. Set the initial time $t = 1$
2. Capture the current *bin value variation* as the current state $q_t = |\tilde{h}_i(t) - \tilde{h}_i(t-1)|$
3. Capture the current *bin value* as the current symbol $O_t = \tilde{h}_i(t)$ observed at state q_t
4. If $(t = T_i)$ then **Terminate**, else set $t = t + 1$ to make a transition to the next bin and **go to** step 2.

From Fig. 4, one can deduce that the number of symbols of the HMM associated with $\widehat{H}$ is $(N + 1)$ because these symbols are extracted from the set of slices $\{v_0, v_1, \ldots, v_N\}$ obtained after splitting the interval [0, 100]. In a similar way, one can deduce that the number of hidden states of this HMM is also $(N + 1)$ because the maximum state is N and the minimum state is 0. When this algorithm is applied on all the content of $\tilde{H}$, the set $\Delta_{\widehat{H}} = \{\delta_{\widehat{h}_1}, \ldots, \delta_{\widehat{h}_M}\}$ of generated Markov chains is obtained.

$$\begin{array}{ccccccc} \tilde{h}_i(1) & \ldots & \tilde{h}_i(t) & \ldots & \tilde{h}_i(T_i) \\ \uparrow & \vdots & \uparrow & \vdots & \uparrow \\ |\tilde{h}_i(1) - \tilde{h}_i(0)| \rightarrow \ldots \rightarrow & & |\tilde{h}_i(t) - \tilde{h}_i(t-1)| & \rightarrow \ldots \rightarrow & |\tilde{h}_i(T_i) - \tilde{h}_i(T_i - 1)| \end{array}$$

Fig. 4 Generated Markov chain $\delta_{\widehat{h}_i}$ of $\widehat{h}_i$

3.4.3 Example of histogram transformation

Consider the set $H = \{h\}$ composed of only one histogram whose bin values are listed in Table 2. When the former principle is applied on h for $N = 10$, the histograms $\widehat{h}$ and $\tilde{h}$ presented in Table 2 are obtained. Then, the generated Markov chain $\delta_{\widehat{h}}$ of $\widehat{h}$ presented in Fig. 5 is derived.

3.5 HMM training

3.5.1 Construction of the initial HMM

Consider a set H of histograms and its associated set of normalized histograms $\widehat{H}$. In this work, the parameters of the initial HMM $\lambda^0_{\widehat{H}}$ associated with $\widehat{H}$ are set in such a way that they statistically capture the states transitions and the observation symbols probabilities distributions from the content of $\Delta_{\widehat{H}}$. These parameters are set as follows:

1. Its number of states is $(N + 1)$, where N is the user-defined integer used to split the interval [0, 100]. The set of states is $S = \{0, 1, 2, \ldots, N\}$.
2. Its number of observation symbols is also $(N + 1)$, and the set of symbols is $V = \{0, 1, 2, \ldots, N\}$.
3. Its probability of transiting from state j to state k is calculated in Eq. 9 where transit$(j, k, \Delta_{\widehat{H}})$ is the number of transitions from state j to state k in $\Delta_{\widehat{H}}$ and

$$\begin{array}{lccccccccc} \textit{Values}: & 3 & 7 & 10 & 9 & 2 & 8 & 4 & 0 & 10 \\ & \uparrow & \uparrow & \uparrow & \uparrow & \uparrow & \uparrow & \uparrow & \uparrow & \uparrow \\ \textit{Variations}: & 3 \rightarrow & 4 \rightarrow & 3 \rightarrow & 1 \rightarrow & 7 \rightarrow & 6 \rightarrow & 4 \rightarrow & 4 \rightarrow & 10 \end{array}$$

Fig. 5 Generated Markov chain $\delta_{\widehat{h}}$ of $\widehat{h}$

Table 2 Bin values of h, $\widehat{h}$ and $\tilde{h}$ when $N = 10$

Index	0	1	2	3	4	5	6	7	8	9
h		2	5	8	7	1	6	3	0	8
$\widehat{h}$		25	62.5	100	87.5	12.5	75	37.5	0	100
$\tilde{h}$	0	3	7	10	9	2	8	4	0	10

transit$(j,-,\Delta_{\widehat{H}})$ is the number of transitions from state j to any destination in $\Delta_{\widehat{H}}$.

$$A^0_{\widehat{H}}[j,k] = \frac{\text{transit}(j,k,\Delta_{\widehat{H}})}{\text{transit}(j,-,\Delta_{\widehat{H}})+1}. \quad (9)$$

4. Its probability of observing symbol k at state j is calculated in Eq. 10 where observe$(k,j,\Delta_{\widehat{H}})$ is the number of times where symbol k is observed at state j in $\Delta_{\widehat{H}}$, and observe$(-,j,\Delta_{\widehat{H}})$ is the number of occurrences of state j in $\Delta_{\widehat{H}}$, whatever is the symbol observed.

$$B^0_{\widehat{H}}[j,k] = \frac{\text{observe}(k,j,\Delta_{\widehat{H}})}{\text{observe}(-,j,\Delta_{\widehat{H}})+1}. \quad (10)$$

5. Its probability that a sequence starts with state j is calculated in Eq. 11 where start$(j,\Delta_{\widehat{H}})$ is the number of elements in $\Delta_{\widehat{H}}$ starting with state j.

$$\pi^0_{\widehat{H}}[j] = \frac{\text{start}(j,\Delta_{\widehat{H}})}{|\Delta_{\widehat{H}}|+1}. \quad (11)$$

The parameters of $\lambda^0_{\widehat{H}}$ are not probability distributions due to the 1 added to the denominator of their components. We intentionally introduced this error to avoid eventual divisions by zero in $A^0_{\widehat{H}}$ and $B^0_{\widehat{H}}$. This also enabled us to avoid zero probabilities in $\pi^0_{\widehat{H}}$. Similarly to what was done in [34], this error is solved here by equitably redistributing in each line the missing quantity. The readjusted model $\lambda^1_{\widehat{H}} = (A^1_{\widehat{H}}, B^1_{\widehat{H}}, \pi^1_{\widehat{H}})$ is obtained as follows:

1. $A^1_{\widehat{H}}[j,k] = A^0_{\widehat{H}}[j,k] + \frac{1}{N+1}\left(1 - \sum_{l=1}^{N+1} A^0_{\widehat{H}}[j,l]\right)$,
2. $B^1_{\widehat{H}}[j,k] = B^0_{\widehat{H}}[j,k] + \frac{1}{N+1}\left(1 - \sum_{l=1}^{N+1} B^0_{\widehat{H}}[j,l]\right)$,
3. $\pi^1_{\widehat{H}}[j] = \pi^0_{\widehat{H}}[j] + \frac{1}{N+1}\left(1 - \sum_{l=1}^{N+1} \pi^0_i[l]\right)$.

3.5.2 The training phase

If $(|\widehat{H}| = 1)$, then the readjusted initial HMM $\lambda^1_{\widehat{H}}$ of $\widehat{H}$ is trained for one single sequence, and otherwise $\lambda^1_{\widehat{H}}$ is trained for multiple sequences. In both situations, the training sequences are composed of observation symbols (bin values) in $\Delta_{\widehat{H}}$. If we consider the set $H = \{h\}$ used in the example of Sect. 3.4.3, then the initial HMM $\lambda^1_{\widehat{H}}$ of $\widehat{H}$ will be trained with the following sequence of symbols: 3, 7, 10, 9, 2, 8, 4, 0, 10. The result of the training is the final HMM $\lambda_{\widehat{H}}$ associated with $\widehat{H}$.

3.6 Normalized similarity rate

3.6.1 Definition

Consider two finite sets H_1 and H_2 of histograms. We define in Eq. 12 the *normalized similarity rate* $\widehat{\sigma}(H_1,H_2)$ between H_1 and H_2 as the probability that the HMMs $\lambda_{\widehat{H}_1}$ and $\lambda_{\widehat{H}_2}$ associated with $\widehat{H}_1$ and $\widehat{H}_2$ generate identical observation sequences. In our context, this measure evaluates the certainty rate that $\lambda_{\widehat{H}_1}$ and $\lambda_{\widehat{H}_2}$ generate other very near histograms.

$$\widehat{\sigma}(H_1,H_2) = 100 \times \text{Sim}(\lambda_{\widehat{H}_1}, \lambda_{\widehat{H}_2}) \quad (\text{in } \%). \quad (12)$$

3.6.2 Example of computation

Consider the three sets of histograms $H_i = \{h_i\}$ with $(1 \le i \le 3)$ where $h_1 = [2,5,8,7,1,6,3,0,8]$ (the histogram used in Sect. 3.4.3), $h_2 = [15,75,45,60,30,15,15,45,75,30]$ and $h_3 = [5,25,15,20,10,5,5,15,25,10,20,25]$. Although h_2 and h_3 seem to be very different at the first sight, their normalized histograms are quite identical. In fact, their 10 first bins values are identical, and only the two last bins of $\widehat{h}_3$ are nonexistent in $\widehat{h}_2$. Therefore, our expectation is that $\widehat{\sigma}(H_1,H_2)$ and $\widehat{\sigma}(H_1,H_3)$ must have quite identical values, given that $\widehat{h}_2$ and $\widehat{h}_3$ are quite similar.

The normalized similarity rates $\widehat{\sigma}(H_1,H_2)$, $\widehat{\sigma}(H_1,H_3)$ and $\widehat{\sigma}(H_2,H_3)$ have been calculated for 19 values of N taken between 10 and 100 with a step of 5, and the results are presented in Fig. 6a. During this experience, the *Baum–Welch* algorithm was applied with a maximum number of iterations equal to 500, and this maximum value will be used in all the examples of this paper. All the experiments realized in this paper were executed on a personal computer with the following properties: (1) Processors: *Intel(R) Core(TM) i7-2670QM CPU @2.2GHz 2.2GHz*, (2) RAM: *8 GB*. Figure 6a experimentally reveals that $\widehat{\sigma}$ is unstable for values of $(N < 50)$ and becomes very stable when $(N \ge 50)$. For this reason, values of $(N \ge 50)$ are recommended, and in the rest of this paper, the value $N = 50$ is adopted. It can also be observed that $\widehat{\sigma}$ effectively captures the fact that $\widehat{h}_2$ and $\widehat{h}_3$ are quite identical because the values of $\widehat{\sigma}(H_2,H_3)$ vary between 98.85 and 99.41% when $(N \ge 50)$. Figure 6a also shows that our former expectation is fulfilled because the curves associated with $\widehat{\sigma}(H_1,H_2)$ and $\widehat{\sigma}(H_1,H_3)$ are quite identical when $(15 \le N \le 100)$. The time costs in seconds of this experience are presented in Fig. 6b. This figure shows that when $(N < 50)$, the computation of $\widehat{\sigma}$ takes less than 0.5 s. But when $(N \ge 50)$, this time cost gradually augmented for each pair, almost reaching 4 s for the pair (H_2,H_3).

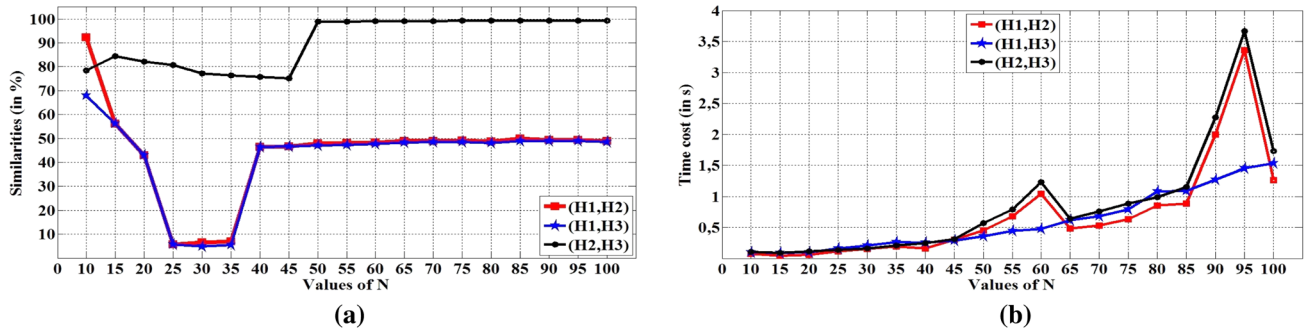

Fig. 6 Values and time costs of $\widehat{\sigma}(H_1, H_2)$, $\widehat{\sigma}(H_1, H_3)$ and $\widehat{\sigma}(H_2, H_3)$ when N varies between 10 and 100. **a** Values of the normalized similarity rates. **b** Time costs of the normalized similarity rates

3.7 The amplitude coefficient

At the beginning of the proposed methodology, we have normalized the histograms. Now, to take the histograms amplitudes into account in the similarity computation, the normalized similarity rate must be weighted by a suitable amplitude coefficient. In order to determine the value of the amplitude coefficient, the expression $\widehat{\sigma}(H_1, H_2) = 100\%$ must be well interpreted. This expression means that $(\widehat{H_1} \approx \widehat{H_2})$; the accuracy of this approximation depends on the value of N. In the limit case where $(\widehat{H_1} = \widehat{H_2})$, this expression means that for each histogram $h_1 \in H_1$, there exists a unique histogram $h_2 \in H_2$ verifying $(\widehat{h_1} = \widehat{h_2})$. As it is proved in Eq. 13, $(\widehat{h_1} = \widehat{h_2})$ implies that there exist two positive coefficients θ and θ' such that $(h_1 = \theta h_2)$ and $(h_2 = \theta' h_1)$.

$$\begin{aligned}(\widehat{h_1} = \widehat{h_2}) &\Leftrightarrow \left(\frac{100}{H_{1\max}} \times h_1 = \frac{100}{H_{2\max}} \times h_2\right)\\ &\Leftrightarrow h_1 = \left(\frac{H_{1\max}}{H_{2\max}}\right) h_2\\ &\Leftrightarrow h_1 = \theta h_2 \ \text{ where } \ \theta = \left(\frac{H_{1\max}}{H_{2\max}}\right)\\ &\Leftrightarrow h_2 = \theta' h_1 \ \text{ where } \ \theta' = \left(\frac{H_{2\max}}{H_{1\max}}\right).\end{aligned} \tag{13}$$

Equation 13 enables us to derive the symmetric *amplitude coefficient* $\theta(H_1, H_2)$ computed in Eq. 14. According to this equation, $\theta(H_1, H_2)$ is close to 0 when the histograms in H_1 and H_2 have distant amplitudes. In the same way, $\theta(H_1, H_2)$ is close to 1 when the amplitudes of the histograms in H_1 and H_2 are near.

$$\theta(H_1, H_2) = \frac{\min\left(H_{1\max}, H_{2\max}\right)}{\max\left(H_{1\max}, H_{2\max}\right)}. \tag{14}$$

3.8 Similarity rate computation

We propose in Eq. 15 to compute the symmetric *similarity rate* $\sigma(H_1, H_2)$ between two finite sets H_1 and H_2 of histograms by calculating their normalized similarity rate $\widehat{\sigma}(H_1, H_2)$, weighted by their amplitude coefficient $\theta(H_1, H_2)$.

$$\sigma(H_1, H_2) = \theta(H_1, H_2) \times \widehat{\sigma}(H_1, H_2) \quad (\text{in } \%). \tag{15}$$

Equation 15 produces the following similarity rates between the three sets H_1, H_2 and H_3 taken as examples in Sect. 3.6.2:

- $\sigma(H_1, H_2) = \frac{\min(8,75)}{\max(8,75)} \times 47.79 = \frac{8}{75} \times 47.79 = 5.10\%$
- $\sigma(H_1, H_3) = \frac{\min(8,25)}{\max(8,25)} \times 47.04 = \frac{8}{25} \times 47.04 = 15.05\%$
- $\sigma(H_2, H_3) = \frac{\min(25,75)}{\max(25,75)} \times 98.85 = \frac{25}{75} \times 98.85 = 32.95\%.$

In identical experimental conditions, we have compared the three histograms of Fig. 1a–c located in "Introduction" of this paper. The following results we obtained reveal that h_3 has the same similarity rate with h_1 and h_2 unlike what was expressed by the *Euclidean distance*.

- $\sigma(\{h_1\}, \{h_2\}) = 51.95\%$,
- $\sigma(\{h_1\}, \{h_3\}) = 31.86\%$,
- $\sigma(\{h_2\}, \{h_3\}) = 31.86\%$.

3.9 Properties of $\widehat{\sigma}$ and σ

The first property that we state in this paper is related to the true nature of σ which is only a similarity measure but not a metric.

Property 1 *σ is a similarity measure but not a metric.*

Proof $\sigma(H_1, H_2)$ is a similarity measure because it is derived from $\text{Sim}(\lambda_{\widehat{H_1}}, \lambda_{\widehat{H_2}})$ which is a similarity measure [33]. But

σ is not a metric because it does not verify triangular inequality. Indeed, in the first example of Sect. 3.8 we have $\sigma(H_2, H_3) > \sigma(H_2, H_1) + \sigma(H_1, H_3)$ because $32.95 > (5.10 + 15.05)$. □

The immediate consequence of Property 1 is that σ can not be efficiently used to enhance the accuracy of metrics-dependent algorithms like the *K-NN* algorithm in a flat classification process. But, σ can rather be used to initially organize the classes into a hierarchical structure (a taxonomy) according to their similarities, before to perform hierarchical classification with common basic classifiers.

Let us present now properties of $\widehat{\sigma}$ and σ when the amplitudes of their inputs sets of histograms are modified. Consider a set $H = \{h_1, \ldots, h_M\}$ of histograms and a positive real number x. If we note $(x.H) = \{x.h_1, \ldots, x.h_M\}$ the set containing all the histograms of H with their bin values multiplied by x, then the following properties are verified:

Property 2 *Consider two finite sets H_1 and H_2 of histograms. For all positive real numbers x and y: $\widehat{\sigma}(x.H_1, y.H_2) = \widehat{\sigma}(H_1, H_2)$*

Proof To demonstrate Property 2, we first need to show that $\widehat{H} = \widehat{(x.H)}$ for every finite set H of histograms and for all positive real number x. $\forall x > 0$, $\forall h \in H$ and $\forall (1 \leq j \leq |h|)$, Eq. 16 proves that $\widehat{H} = \widehat{(x.H)}$.

$$\begin{cases} \widehat{h}(j) = h(j) \times \left(\frac{100}{H_{\max}}\right) & ① \\ \widehat{(x.h)}(j) = x.h(j) \times \left(\frac{100}{x.H_{\max}}\right) = h(j) \times \left(\frac{100}{H_{\max}}\right) & ② \end{cases}$$

$$\begin{aligned} ① = ② \;&\Leftrightarrow\; \forall x > 0, \forall h \in H, \forall (1 \leq j \leq |h|) : \widehat{h}(j) = \widehat{(x.h)}(j) \\ &\Leftrightarrow\; \widehat{H} = \widehat{(x.H)}. \end{aligned} \tag{16}$$

Let us now deduce Property 2. $\widehat{\sigma}(x.H_1, y.H_2)$ is defined as the similarity rate between $\widehat{(x.H_1)}$ and $\widehat{(y.H_2)}$. According to Eq. 16, $\widehat{H_1} = \widehat{(x.H_1)}$ and $\widehat{H_2} = \widehat{(y.H_2)}$. We can consequently deduce that $\widehat{\sigma}(x.H_1, y.H_2)$ is also the similarity rate between $\widehat{H_1}$ and $\widehat{H_2}$. However, $\widehat{\sigma}(H_1, H_2)$ is basically defined as the similarity rate between $\widehat{H_1}$ and $\widehat{H_2}$. Therefore, we can conclude that: $\widehat{\sigma}(x.H_1, y.H_2) = \widehat{\sigma}(H_1, H_2)$ □

Property 3 *Consider a finite set H of histograms. For all positive real numbers x and y, we have:*

$$\sigma(x.H, y.H) = \frac{\min(x, y)}{\max(x, y)} \times 100 \quad \text{(in \%)}.$$

Proof Equation 17 demonstrates Property 3.

$$\begin{aligned} \sigma(x.H, y.H) &= \theta(x.H, y.H) \times \widehat{\sigma}(x.H, y.H) \\ &= \theta(x.H, y.H) \times \widehat{\sigma}(H, H) \,(\textit{Cf. Property}\, 2) \\ &= \theta(x.H, y.H) \times 100 \\ &= \frac{\min\left(x.H_{\max}, y.H_{\max}\right)}{\max\left(x.H_{\max}, y.H_{\max}\right)} \times 100 \\ &= \frac{H_{\max} \times \min(x, y)}{H_{\max} \times \max(x, y)} \times 100 \\ &= \frac{\min(x, y)}{\max(x, y)} \times 100 \end{aligned} \tag{17}$$

□

Property 4 *Consider two finite sets H_1 and H_2 of histograms. For all positive real numbers x and y: $\sigma(x.H_1, y.H_2) = \sigma(H_1, \frac{y}{x}.H_2) = \sigma(\frac{x}{y}.H_1, H_2)$*

Proof Equation 18 demonstrates that $\sigma(x.H_1, y.H_2) = \sigma(H_1, \frac{y}{x}.H_2)$. A similar reasoning enables to demonstrate that $\sigma(x.H_1, y.H_2) = \sigma(\frac{x}{y}.H_1, H_2)$ by factorizing y instead of x at the 3^{rd} line of Eq. 18. If $x = y$, then Property 4 becomes: $\sigma(x.H_1, x.H_2) = \sigma(H_1, H_2)$.

$$\begin{aligned} \sigma(x.H_1, y.H_2) &= \theta(x.H_1, y.H_2) \times \widehat{\sigma}(x.H_1, y.H_2) \\ &= \frac{\min\left(x.H_{1\max}, y.H_{2\max}\right)}{\max\left(x.H_{1\max}, y.H_{2\max}\right)} \times \widehat{\sigma}(x.H_1, y.H_2) \\ &= \frac{x \times \min\left(H_{1\max}, \frac{y}{x}.H_{2\max}\right)}{x \times \max\left(H_{1\max}, \frac{y}{x}.H_{2\max}\right)} \times \widehat{\sigma}(x.H_1, y.H_2) \\ &= \theta\left(H_1, \frac{y}{x}.H_2\right) \times \widehat{\sigma}(x.H_1, y.H_2) \\ &= \theta\left(H_1, \frac{y}{x}.H_2\right) \times \widehat{\sigma}(H_1, H_2) \,(\text{Cf. Property2}) \\ &= \theta\left(H_1, \frac{y}{x}.H_2\right) \times \widehat{\sigma}\left(H_1, \frac{y}{x}.H_2\right) \,(\text{Cf. Property2}) \\ &= \sigma(H_1, \frac{y}{x}.H_2). \end{aligned} \tag{18}$$

□

Since $\widehat{\sigma}$ and σ are functions that manipulate finite sets, it is crucial to describe their behaviors when they are applied to the empty set. It is in this perspective that we have adopted the two following conventions:

Property 5 $\widehat{\sigma}(\emptyset, \emptyset) = \sigma(\emptyset, \emptyset) = 100\%$ *because the empty set is naturally completely similar to itself.*

Property 6 *For every finite set of histograms, $H \neq \emptyset$ we have: $\widehat{\sigma}(H, \emptyset) = \sigma(H, \emptyset) = 0\%$ because the empty set is naturally completely different from any non-empty set.*

3.10 Time cost of σ

The computation of σ involves two HMMs training phases that may consume a lot of time. Consider a set $H = \{h_1, \ldots, h_m\}$ of histograms. In order to obtain the HMM $\lambda_{\widehat{H}}$, the duration of the training phase depend on the following elements:

1. The number $|H|$ of histograms contained in H.
2. The number $|h_i|$ of bins of each histogram $h_i \in H$.
3. The shapes of each histogram $h_i \in H$.
4. The user-defined maximum number of iterations for the *Bauwm–Welch* algorithm (500 in this work).

Therefore, almost all the computation time of σ is dedicated to the HMMs training phases in such a way that, the time used to compute the amplitude coefficient and the final similarity result can easily be neglected. For this reason, only the time costs of the HMMs training phases will be presented in the rest of this paper.

4 Experimental results

4.1 Color image comparison

σ can be used to define the color similarity rate (CSR) between two sets of color images. A survey of methods for color image indexing and retrieval in image databases is realized in [35]. Unlike existing approaches that can only compare two color images, we rather propose here to compare two sets of color images. We first show how to compute the CSR of two single color images, and then, we generalize to sets of color images. Before to achieve this goal, let us exhibit the relationship between color images and histograms.

4.1.1 Color histogram of an image

In image processing, the distribution of colors in a color image for each dimension of a color space can be represented by a histogram-based local descriptor called *color histogram* or *histogram of colors*. In this section, the *RGB* color space is selected. In order to construct color histograms composed of s bins in the *RGB* color space, each primary color $p \in \{R, G, B\}$ is first sampled into s ranges of intensities $\{i_1, i_2, \ldots, i_s\}$. Thereafter, the color histogram h_p of an image I in each dimension p is constructed in such a way that the value of the kth bin of this histogram is the number of pixels in I that have the color p with an intensity $i \in i_k$. Generally, $s \in \{64, 128, 256, \ldots\}$ and in this section, the value $s = 128$ has been selected.

4.1.2 Color similarity rate between two images

Color histograms suffer from the lack of spatial information; therefore, they cannot differentiate patterns of colors [18]. Nevertheless, they can still be used to evaluate the overall color similarity rate between two images as it is demonstrated in this section. Consider two color images I and I', respectively, represented in the RGB color space by their color histograms (h_R, h_G, h_B) and (h'_R, h'_G, h'_B). If we set $H_p = \{h_p\}$ and $H'_p = \{h'_p\}$ for each dimension $p \in \{R, G, B\}$, then the similarity rate per dimension between I and I' can first be evaluated by applying σ on their corresponding color histograms in each dimension. As presented in Eq. 19, this outputs a point $\Omega_{I,I'}$ of the affine space $\mathbb{R}^3$ whose components are always in [0, 100]. In the rest of this paper, we will designate by $\Omega^n_{\max}$ the point of the affine space $\mathbb{R}^n$ whose components are all equal to 100.

$$\Omega_{I,I'} = \begin{bmatrix} \sigma(H_R, H'_R) \\ \sigma(H_G, H'_G) \\ \sigma(H_B, H'_B) \end{bmatrix}. \tag{19}$$

Unlike other existing distance measures between histograms, σ has the advantage to have an upper bound which is 100%. Given that the highest similarity rate between I and I' in each dimension is 100%, the distance between I and I' can therefore be seen as the straight-line distance separating the point $\Omega_{I,I'}$ from the point $\Omega^3_{\max} = [100, 100, 100]$ representing the maximum possible similarities between any pair of images. From this observation, we propose in Eq. 20 to consider the distance $d_\sigma(I, I')$ between I and I' as the *Euclidean distance* between $\Omega_{I,I'}$ and $\Omega^3_{\max}$.

$$\begin{aligned} d_\sigma(I, I') &= L_2(\Omega^3_{\max}, \Omega_{I,I'}) \\ &= \sqrt{\sum_{p \in \{R,G,B\}} (100 - \sigma(H_p, H'_p))^2}. \end{aligned} \tag{20}$$

It is obvious that if the color histograms of I and I' are strictly identical in each dimension, then $d_\sigma(I, I') = 0$. On the other hand, if the color histograms of I and I' are completely different in each dimension (i.e., $\Omega_{I,I'} = [0, 0, 0]$), then Eq. 20 gives $\sqrt{30000} = 100\sqrt{3}$. This value is the upper bound of the proposed distance between two color images in the RGB space. When we divide $d_\sigma(I, I')$ by this upper bound, we obtain a dissimilarity coefficient. Therefore, the dissimilarity rate between I and I' gives

$100 \times \left(\frac{d_\sigma(I,I')}{100\sqrt{3}}\right) = \frac{d_\sigma(I,I')}{\sqrt{3}}$ (in %). This allows us to define in Eq. 21 the CSR between I and I' noted $\chi(I, I')$.

$$\chi(I, I') = 100 - \frac{d_\sigma(I, I')}{\sqrt{3}} \qquad (in\ \%). \tag{21}$$

4.1.3 Example of CSR computation

The proposed CSR has been calculated for the six images of Fig. 7a–f. A human being can obviously observe strong overall color similarities between the pair of images $(I_1, I_2) = \{sky + green\ sea\ with\ stained\ areas\}$,

Fig. 7 The 6 images to be compared. **a** Image I_1. **b** Image I_3. **c** Image I_5. **d** Image I_2. **e** Image I_4. **f** Image I_6

Table 3 CSR between the distinct pairs of images of Fig. 7a–f

I	I'	$\Omega_{I,I'}$			$d_\sigma(I, I')$	$\chi(I, I')$ in %
		R	G	B		
I_1	I_2	25.97	73.87	58.48	88.81	48.72
I_1	I_3	40.25	20.94	35.42	118.29	31.71
I_1	I_4	16.79	11.35	22.40	144.24	16.72
I_1	I_5	41.69	41.86	77.23	85.44	50.67
I_1	I_6	39.21	81.70	69.56	70.40	59.35
I_2	I_3	51.85	29.17	20.37	116.94	32.48
I_2	I_4	54.58	28.90	19.35	116.72	32.61
I_2	I_5	22.83	59.57	65.50	93.70	45.90
I_2	I_6	38.04	57.51	77.70	78.37	54.75
I_3	I_4	41.84	34.84	33.04	110.06	**36.46**
I_3	I_5	56.32	33.92	21.47	111.54	35.61
I_3	I_6	54.11	29.67	16.77	118.23	31.74
I_4	I_5	26.07	29.78	26.52	125.68	27.44
I_4	I_6	23.63	30.46	11.67	135.91	21.53
I_5	I_6	50.38	59.86	33.09	92.47	46.61

The bold value is the CSR of the pair that fulfilled the conjecture

Table 4 Time costs in seconds of the HMMs training phases for each image $I_1, \ldots, I_6$ of Fig. 7a–f

	I_1	I_2	I_3	I_4	I_5	I_6
H_R	1.36	2.89	5.63	13.90	3.82	2.62
H_G	6.54	14.02	4.70	14.09	2.63	4.14
H_B	3.78	6.91	7.25	4.31	9.25	4.28

$(I_3, I_4) = \{sky + houses\ made\ of\ sand\}$ and $(I_5, I_6) = \{sky + snow + black\ mountain\}$. Therefore, our conjecture is that $\chi(I_1, I_2)$ must be the highest CSR between any pair of distinct images including I_1 or I_2. The same reasoning allows us to conjecture that $\chi(I_3, I_4)$ and $\chi(I_5, I_6)$ should be the highest CSR between any pair of distinct images including their respective targeted images. We calculated the CSR between all the possible pairs of distinct images in $\{I_1, \ldots, I_6\}$, and the results are presented in Table 3. Table 4 shows the time consumed during the HMMs training phases for each image.

As it is shown in Table 3, only the pair (I_3, I_4) fulfilled our conjecture. Unfortunately, $\chi(I_1, I_2) = 48.72\%$ and $\chi(I_5, I_6) = 46.61\%$ are both lower than $\chi(I_1, I_5) = 50.67\%$. In order to overcome this limitation, the computation of the proposed CRS between two color images can be enhanced by dividing each image into many regular sub-images before to perform the comparison. The next section is devoted to this issue.

4.1.4 Enhanced computation of the CSR

In order to enhance image comparison, the authors of [18] proposed to break an image into spatially regular sub-images, before to calculate the color histograms of each sub-image. This principle is applied here to break the image I_1 of Fig. 7a into 16 regular sub-images using a 4×4 grid, and the result is presented in Fig. 8. Each sub-image is indexed by a pair (j, k) of labels, respectively, located on the left and below the image.

When this procedure is applied on any image I with a $n \times n$ grid, the image becomes a set $I = \{I_{j,k} | 1 \leq j, k \leq n\}$ composed of n^2 sub-images. Consider now two color images I and I', each broken into n^2 sub-images using a $n \times n$ grid according to this principle to produce the sets $I = \{I_{j,k} | 1 \leq j, k \leq n\}$ and $I' = \{I'_{j,k} | 1 \leq j, k \leq n\}$. Equations 20 and 21 are still applied on the two sets I and I' to, respectively, compute the distance between them and their similarity rate. The only difference here is that for each $p \in \{R, G, B\}$, the sets H_p and H'_p of color histograms are no longer singletons as it was previously the case. Each set is now composed of n^2 color histograms, i.e., $H_p = \{h_{j,k,p} | 1 \leq j, k \leq n\}$ and $H'_p = \{h'_{j,k,p} | 1 \leq j, k \leq n\}$. The histograms $h_{j,k,p}$ and $h'_{j,k,p}$ are, respectively, the color histograms in the dimension p associated with the sub-image of index (j, k) in I and I'. This implies that the *Baum–Welch* algorithm will be executed for multiple sequences during the computation of the three components of $\Omega_{I,I'}$.

After breaking each image of Fig. 7a–f into n^2 sub-images using a $n \times n$ grid with $n = 2, 3, 4, 5$ (these are the values used in [18]), we have once again computed the CSR between all the distinct pairs of this set of images. The results are presented in Table 5a–d. Although our three aforementioned conjectures were just partially fulfilled for $n = 2, 3, 4$, they were all satisfied for $n = 5$ as it can be observed in Table 5d.

We measured the time consumed during the HMMs training phases for each image when $n = 5$ because it is the only case where all our three conjectures were satisfied. We experimentally observed that each HMM training phase roughly took between 9 and 13 min.

4.1.5 CSR between two sets of images

Consider two sets of color images $E = \{I_1, \ldots, I_{|E|}\}$ and $E' = \{I'_1, \ldots, I'_{|E'|}\}$. The CSR between the sets E and E' is computed as follows: For each primary color $p \in \{R, G, B\}$, the two sets of color histograms $H_p = \{h_{k,p} | 1 \leq k \leq |E|\}$ and $H'_p = \{h'_{k,p} | 1 \leq k \leq |E'|\}$, respectively, associated with the images in E and E' are first constructed. Finally, Eqs. 20 and 21 are, respectively, used to compute the distance and the CSR between E and E'. Similarly to what is done in Sect. 4.1.4, one can initially decide to break each image into n^2 sub-images, given a user-defined value of n in order to compute the enhanced CSR.

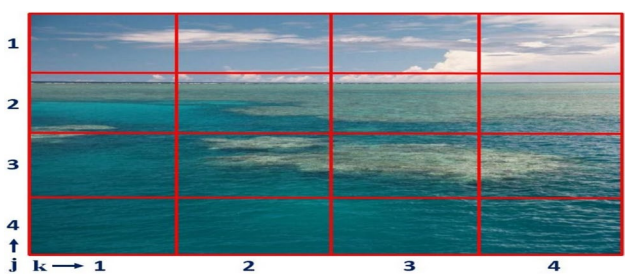

Fig. 8 Image I_1 of Fig. 7a broken into 16 sub-images using a 4×4 grid. The labels (j, k) allow to index the sub-images

4.2 Comparison of functions curves

A γ-dimensional real function is a function with $(\gamma - 1)$ real variables, with $\gamma = 2, 3, 4, \ldots$. In this paper, only the values $\gamma = 2$ and $\gamma = 3$ are considered, and in this context, the curve of a γ-dimensional real function generally expresses the evolution of a specific phenomenon/process according to the variations of $(\gamma - 1)$ variables. Therefore, comparing the curves of two γ-dimensional real functions in this context enables to compare the evolution of their associated phenomena/processes. Unlike existing approaches that generally determine the relative positions

Table 5 Enhanced CSR of the distinct pairs of images of Fig. 7a–f

I	I'	$\Omega_{I,I'}$			$d_\sigma(I,I')$	$\chi(I,I')$ in %
		R	G	B		
(a) Enhanced CSR using a 2×2 grid						
I_1	I_2	29.84	69.46	61.48	85.66	50.54
I_1	I_3	28.76	24.42	27.66	126.57	26.92
I_1	I_4	29.01	30.24	18.96	128.35	25.90
I_1	I_5	65.27	41.93	80.25	70.48	59.31
I_1	I_6	52.07	40.19	42.51	95.81	44.68
I_2	I_3	69.37	11.35	12.14	128.51	25.80
I_2	I_4	30.74	13.64	8.52	143.61	17.09
I_2	I_5	44.51	56.10	62.54	80.06	53.78
I_2	I_6	32.07	56.08	35.86	103.23	40.40
I_3	I_4	35.57	37.63	62.68	97.12	**43.93**
I_3	I_5	42.14	28.53	30.86	115.05	33.58
I_3	I_6	49.49	31.47	26.59	112.41	35.10
I_4	I_5	23.63	32.43	12.03	134.67	22.25
I_4	I_6	28.55	21.51	25.81	129.50	25.23
I_5	I_6	49.41	69.71	45.03	80.62	53.46
(b) Enhanced CSR using a 3×3 grid						
I_1	I_2	40.69	56.71	82.59	75.46	**56.43**
I_1	I_3	57.54	35.83	38.30	98.63	43.06
I_1	I_4	18.83	17.58	12.74	144.90	16.34
I_1	I_5	35.82	39.37	45.80	103.60	40.18
I_1	I_6	42.05	47.63	41.79	97.41	43.76
I_2	I_3	46.64	30.07	42.86	104.89	39.44
I_2	I_4	24.92	22.73	8.35	141.44	18.34
I_2	I_5	29.65	39.26	69.59	97.79	43.54
I_2	I_6	36.12	40.27	57.67	97.16	43.90
I_3	I_4	25.47	56.65	31.23	110.29	36.33
I_3	I_5	80.50	47.95	45.99	77.50	55.25
I_3	I_6	76.17	36.20	47.41	86.05	50.32
I_4	I_5	14.80	41.74	19.61	130.83	24.47
I_4	I_6	15.83	38.91	22.01	130.00	24.94
I_5	I_6	53.38	70.16	70.95	62.51	**63.91**
(c) Enhanced CSR using a 4×4 grid						
I_1	I_2	73.08	69.28	39.15	73.29	57.69
I_1	I_3	54.78	18.32	25.41	119.50	31.01
I_1	I_4	29.66	14.53	12.93	140.83	18.69
I_1	I_5	47.01	56.18	49.29	85.44	50.67
I_1	I_6	58.03	53.35	35.63	89.90	48.10
I_2	I_3	48.65	21.48	26.85	118.96	31.32
I_2	I_4	20.46	17.99	20.20	139.36	19.54
I_2	I_5	51.01	69.41	70.60	64.81	62.58
I_2	I_6	59.02	68.70	58.72	66.05	61.87
I_3	I_4	46.23	74.49	44.53	81.36	**53.03**
I_3	I_5	38.36	27.62	42.51	111.10	35.86
I_3	I_6	45.19	25.09	43.11	108.86	37.15
I_4	I_5	19.96	22.94	21.83	135.85	21.57
I_4	I_6	15.08	20.73	22.27	139.78	19.30
I_5	I_6	65.12	86.28	79.06	42.94	**75.21**

Table 5 (continued)

I	I'	$\Omega_{I,I'}$			$d_\sigma(I,I')$	$\chi(I,I')$ in %
		R	G	B		
(d) Enhanced CSR using a 5×5 grid						
I_1	I_2	68.30	64.00	65.76	58.94	**65.97**
I_1	I_3	52.30	22.27	21.02	120.65	30.34
I_1	I_4	37.26	14.44	13.67	136.79	21.03
I_1	I_5	45.74	52.82	48.30	88.57	48.87
I_1	I_6	41.48	51.93	44.12	94.12	45.66
I_2	I_3	43.81	23.85	27.13	119.44	31.04
I_2	I_4	22.43	15.80	11.76	144.54	16.55
I_2	I_5	47.07	61.76	78.51	68.74	60.31
I_2	I_6	48.05	59.80	73.60	70.79	59.13
I_3	I_4	37.49	60.43	57.85	85.15	**50.84**
I_3	I_5	30.74	29.74	29.37	121.34	29.95
I_3	I_6	36.33	30.56	29.68	117.56	32.13
I_4	I_5	20.06	19.73	18.45	139.58	19.41
I_4	I_6	21.76	20.28	16.90	139.22	19.62
I_5	I_6	77.71	64.02	69.15	52.38	**69.76**

Each image is broken into n^2 sub-images using a $n \times n$ grid, with $n = 2, 3, 4, 5$. Bold values are the CSR of the pairs that fulfilled the conjectures

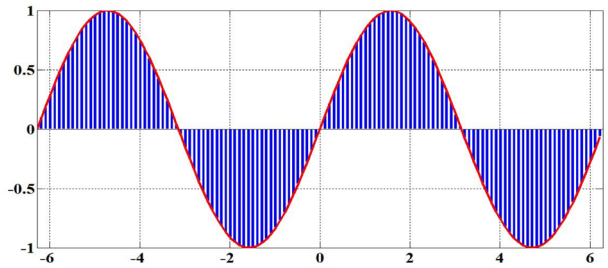

Fig. 9 Bar diagram resulting from the sampling of $f(x) = \sin(x)$ in the interval $\rho = [-2\pi, 2\pi]$

(above/below) of the curves, we rather want to measure the overall similarity rate between the visual shapes of these curves.

Consider two finite sets W and Z, each containing the curves associated with γ-dimensional real functions, with $\gamma = 2$ or $\gamma = 3$. We have used σ to measure the similarity rate ψ_γ between the visual shapes of the curves contained in W and Z. In this work, only functions whose variables belong to unions of bounded intervals are considered. Let us now show how to compute ψ_2 before to recursively compute ψ_3 based on ψ_2. In the rest of this paper, the acronym 'γD' stands for '*γ-dimensional*.'

4.2.1 Computation of ψ_2

Consider two 2D functions f and f' , respectively, continuous in the intervals ρ and ρ'. In order to measure the similarity rate between the shapes of the curves associated with these two functions, the idea here is to sample f in ρ and f' in ρ' with a fixed sampling step and to compare the resulting bar diagrams. The major difficulty here is that there may exist real numbers $x_1 \in \rho$ or $x_2 \in \rho'$ such that $f(x_1) < 0$ or $f'(x_2) < 0$ as shown in Fig. 9 for $f(x) = \sin(x)$ and $\rho = [-2\pi, 2\pi]$.

Figure 9 proves that the bar diagrams resulting from the sampling of f or f' cannot be considered as histograms in such conditions, because a histogram is a bar diagram exclusively composed of positive bin values. To overcome this difficulty, we propose to lift up or down the curves associated with f and f' in such a way that the lowest point of each curve touches the x axis. The two functions f_ε and f'_ε obtained after lifting the curves of f and f' are formally defined in Eq. 22. According to their definitions, the visual shapes and the relative positions of f_ε and f'_ε are, respectively, identical to those of f and f'. But unlike f and f', when f_ε is sampled in ρ and f'_ε is sampled in ρ', two bar diagrams h and h' with positive bin values (*histograms*) are obtained.

$$\begin{cases} f_\varepsilon(x) = (f(x) - \varepsilon) & \text{and} \\ f'_\varepsilon(x) = (f'(x) - \varepsilon) & \text{where} \\ \varepsilon = \min\big(\min\{f(x) | x \in \rho\}, & \min\{f'(x) | x \in \rho'\}\big). \end{cases} \tag{22}$$

Therefore, the similarity rate between the curves associated with f and f' can be computed as the proposed similarity rate between h and h'. It is important to precise that even when f and f' are positive functions (i.e., $f(x_1) \geq 0$ and

$f'(x_2) \geq 0, \forall x_1 \in \rho$ and $\forall x_2 \in \rho'$), the computation of f_ε and f'_ε is still required in order to have the same basis of histogram calculation. Consider now two sets of 2D functions $W = \{f_1, \ldots, f_n\}$ and $Z = \{f'_1, \ldots, f'_m\}$ where each function f_i is defined in the interval ρ_i with $(1 \leq i \leq n)$, and each function f'_j is defined in the interval ρ'_j with $(1 \leq j \leq m)$. The similarity rate between W and Z is calculated as follows:

1. Select a fixed sampling step. This implicitly determines the sampling frequencies of each function $f_i\,(1 \leq i \leq n)$ and each function $f'_j\,(1 \leq j \leq m)$.
2. Compute ε using Eq. 23.
3. Sample the curves associated with each function $f_{i_\varepsilon}(x) = (f_i(x) - \varepsilon)$ with $(1 \leq i \leq n)$ and each function $f'_{j_\varepsilon}(x) = (f'_j(x) - \varepsilon)$ with $(1 \leq j \leq m)$ at their corresponding sampling frequencies. This will output a histogram h_i corresponding to each function $f_i\,(1 \leq i \leq n)$ and a histogram h'_j corresponding to each function f'_j $(1 \leq j \leq m)$.
4. Use Eq. 24 to compute the similarity rate between W and Z.

$$\varepsilon = \min\left(\min_{1\leq i\leq n}\{f_i(x)|x \in \rho_i\}, \min_{1\leq j\leq m}\{f'_j(x)|x \in \rho'_j\}\right) \quad (23)$$

$$\begin{cases} \psi_2(W, Z) = \sigma(H, H') \quad (\text{in}\%) \text{ where} \\ H = \{h_1, \ldots, h_n\} \text{ and } H' = \{h'_1, \ldots, h'_m\}. \end{cases} \quad (24)$$

Consider, for example, the curves associated with the two 2D functions $f(x) = \sin(x)$ and $f'(x) = \cos(x)$ presented in Fig. 10a with $\rho = \rho' = [-2\pi, 2\pi]$. After calculating $\varepsilon = -1$, the curves associated with $f_\varepsilon(x) = \sin(x) + 1$ and $f'_\varepsilon(x) = \cos(x) + 1$ are presented in Fig. 10b. As it can be observed, the curves in Fig. 10a, b have identical shapes and relative positions. The main difference is that in Fig. 10b, $f_\varepsilon(x) \geq 0$ and $f'_\varepsilon(x) \geq 0$, $\forall x \in [-2\pi, 2\pi]$. We later sample the curves associated with f_ε and f'_ε with a sampling step of 0.05, and the simplified notation $(-2\pi : 0.05 : 2\pi)$ is adopted in this paper to specify sampling frequencies. The resulting histograms h and h' are, respectively, presented in Fig. 10c, d, each histogram being composed of 251 bins. Finally, we calculated the similarity rate between the curves associated with f and f', and we obtained $\psi_2(\{f\}, \{f'\}) = 60.5\%$. The HMMs training phases of h and h' took, respectively, 0.17 and 0.13 s.

4.2.2 Recursive computation of ψ_3

In this section, a recursive algorithm is proposed to compute the similarity rate between two sets of 3D functions. In order to achieve this goal, we assume that we know how to compute the similarity rate between two sets composed of 2D functions using the algorithm presented in Sect. 4.2.1. Let $Z = \{z_1, \ldots, z_n\}$ be a set composed of 3D functions. Each function in Z has the following general expression $z_k = f_k(x_1, x_2)$, with $(1 \leq k \leq n)$, and each variable x_i is defined in the interval $\rho^{z_k}_{x_i}$, with $i \in \{1, 2\}$. For

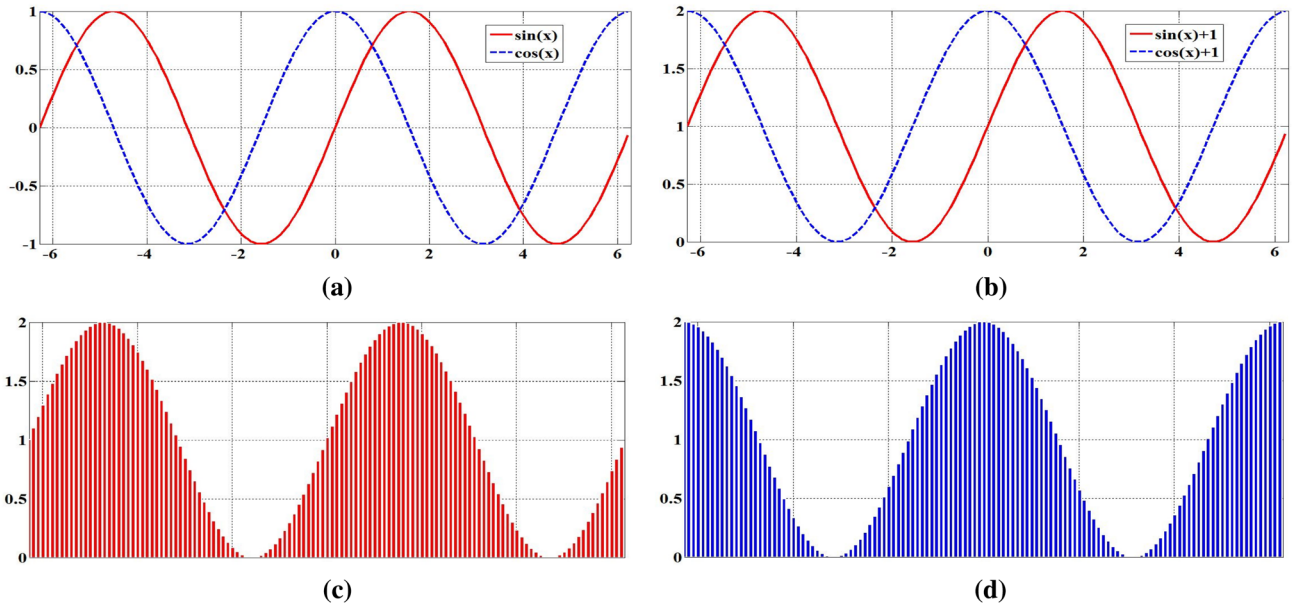

Fig. 10 Transformation of $f(x) = \sin(x)$ and $f'(x) = \cos(x)$ into histograms, both defined in $[-2\pi, 2\pi]$. **a** $f(x) = \sin(x)$ and $f'(x) = \cos(x)$. **b** $f_\varepsilon(x) = \sin(x) + 1$ and $f'_\varepsilon(x) = \cos(x) + 1$. **c** Histogram h derived from $f_\varepsilon(x) = \sin(x) + 1$. **d** Histogram h' derived from $f'_\varepsilon(x) = \cos(x) + 1$

each variable $x_i \in \{x_1, x_2\}$, the set Z can be transformed into a new set Z_{x_i} composed of 2D functions using algorithm 1. In this algorithm, Sample(ρ, k) is a function that samples the interval ρ with a sampling step of k.

Algorithm 1 $From3Dto2D(Z, x_i, step)$

1: $Z_{x_i} \leftarrow \emptyset$
2: **for each** (function $z_k = f_k(x_1, x_2) \in Z$) **do**
3: $I \leftarrow Sample(\rho_{x_i}^{z_k}, step)$
4: **for each** (sample $y \in I$) **do**
5: **if** $(x_i = x_1)$ **then**
6: $Z_{x_i} \leftarrow Z_{x_i} \cup \{f_k(y, x_2)\}$
7: **else**
8: $Z_{x_i} \leftarrow Z_{x_i} \cup \{f_k(x_1, y)\}$
9: **end if**
10: **end for**
11: **end for**
12: **return** Z_{x_i}

Principle of algorithm 1 The new set Z_{x_i} is initialized (line 1), then the set Z is browsed (line 2) and for each function $z_k = f_k(x_1, x_2)$ found in Z, the interval $\rho_{x_i}^{z_k}$ is sampled (line 3). Thereafter, for each sample y resulting from the sampling of $\rho_{x_i}^{z_k}$, a new 2D function is inserted into Z_{x_i}: This is the function z_k where the variable x_i takes the constant value y (lines 4–10). The final result is returned at line 12.

Consider now two sets $W = \{w_1, \ldots, w_m\}$ and $Z = \{z_1, \ldots, z_n\}$ composed of 3D functions. The following principle recursively computes in five steps the similarity rate between W and Z:

1-Selection of the sampling step A user-defined sampling step is selected here.

2-Transformation into sets of 2D functions For each variable $x_i \in \{x_1, x_2\}$, respectively, transform W and Z into the sets W_{x_i} and Z_{x_i} composed 2D functions using algorithm 1.

3-Recursive step Associate with W and Z the point $\Omega^2_{W,Z}$ of $\mathbb{R}^2$ whose coordinate in the ith dimension is the similarity rate between the sets W_{x_i} and Z_{x_i}, with $i \in \{1, 2\}$ as shown in Eq. 25.

$$\Omega^2_{W,Z} = \begin{bmatrix} \psi_2(W_{x_1}, Z_{x_1}) \\ \psi_2(W_{x_2}, Z_{x_2}) \end{bmatrix}. \tag{25}$$

4-Distance computation Given that the highest possible similarity on each dimension is 100%, then the distance $d_\sigma(W, Z)$ between W and Z is computed as the straight-line distance between the point $\Omega^2_{W,Z}$ and the point $\Omega^2_{\max}$ as shown in Eq. 26.

$$\begin{aligned} d_\sigma(W, Z) &= L_2(\Omega^2_{\max}, \Omega^2_{W,Z}) \\ &= \sqrt{\left(100 - \psi_2(W_{x_1}, Z_{x_1})\right)^2 + \left(100 - \psi_2(W_{x_2}, Z_{x_2})\right)^2}. \end{aligned} \tag{26}$$

5-Similarity rate computation When all the components of $\Omega^2_{W,Z}$ are equal to zero, the upper bound of $d_\sigma(W, Z)$ gives $100\sqrt{2}$. Similarly to what is done at Sect. 4.1.2 to derive Eq. 21, we deduce that the similarity rate between W and Z can be calculated with Eq. 27.

$$\psi_3(W, Z) = 100 - \frac{d_\sigma(W, Z)}{\sqrt{2}} \qquad (in\ \%). \tag{27}$$

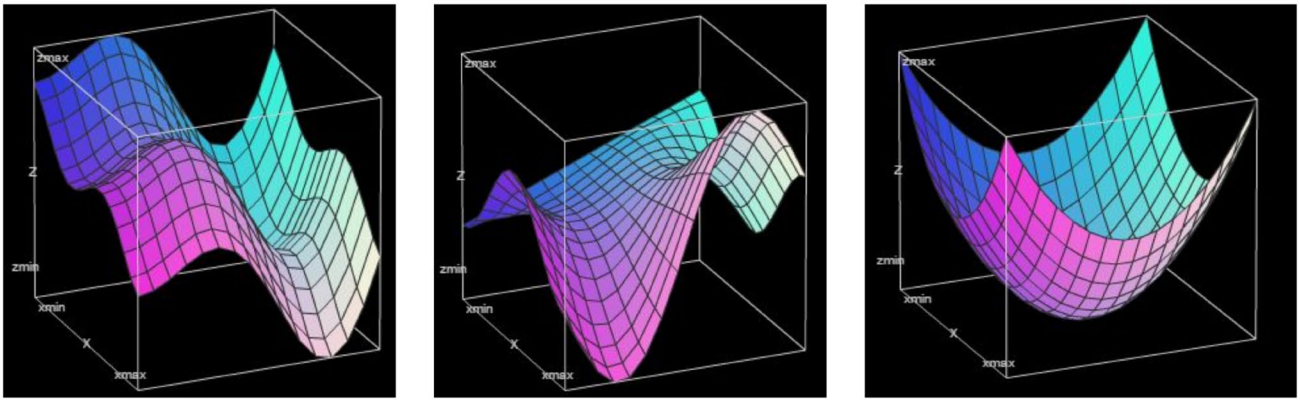

Fig. 11 Images of the curves associated with the three 3D functions f_1, f_2 and f_3 with $\rho_x^{f_1} = [-\pi, \pi]$, $\rho_y^{f_1} = [0, 2\pi]$, $\rho_x^{f_2} = [0, 2\pi]$, $\rho_y^{f_2} = [-\pi, \pi]$ and $\rho_x^{f_3} = \rho_y^{f_3} = [-1, 1]$. **a** $f_1 = x.\cos(x) + y.\sin(y)$. **b** $f_2 = x.\sin(y) + y.\cos(x)$. **c** $f_3 = x^2 + y^2$

Table 6 Similarity rate between the distinct pairs of 3D functions of Fig. 11a–c

W	Z	$\Omega^2_{W,Z}$		$d_\sigma(W,Z)$	$\psi_3(W,Z)$ in %
		x	y		
$\{f_1\}$	$\{f_2\}$	57.64	30.76	81.17	42.60
$\{f_1\}$	$\{f_3\}$	38.25	46.43	81.75	42.20
$\{f_2\}$	$\{f_3\}$	29.96	29.99	99.03	29.98

Table 7 Time consumed during the HMMs training phases of f_1,f_2 and f_3

W	W_2^x	W_2^y
$\{f_1\}$	45′	20′
$\{f_2\}$	22′	37′
$\{f_3\}$	4′	4′

The durations are in minutes

4.2.3 Example of computation of Ω_3

The algorithm presented in Sect. 4.2.2 is applied in this section to measure the similarity rate between 3D functions. Consider the three following 3D functions:

1. $f_1 = x.\cos(x) + y.\sin(y)$ with $\rho_x^{f_1} = [-\pi, \pi]$ and $\rho_y^{f_1} = [0, 2\pi]$ presented in Fig. 11a.
2. $f_2 = x.\sin(y) + y.\cos(x)$ with $\rho_x^{f_2} = [0, 2\pi]$ and $\rho_y^{f_2} = [-\pi, \pi]$ presented in Fig. 11b.
3. $f_3 = x^2 + y^2$ with $\rho_x^{f_3} = \rho_y^{f_3} = [-1, 1]$ presented in Fig. 11c.

Figure 11a–c are drawn with an online available software [36]. At the first sight, it is very hard even for a human being to compare these curves. The algorithm described in Sect. 4.2.2 has been executed on the three distinct pairs of singletons $(\{f_1\}, \{f_2\})$, $(\{f_1\}, \{f_3\})$ and $(\{f_2\}, \{f_3\})$ to compute their similarity rate. The results presented in Table 6 suggest that f_1 and f_2 are the nearest with a similarity rate of 42.6%.

For each singleton $\{f_i\}$ $(1 \leq i \leq 3)$, Table 7 presents the time consumed during its HMMs training phases.

4.3 Automatic taxonomy generation

4.3.1 The proposed methodology

σ has been experimented in automatic taxonomy generation of music genres and color images. The resulting taxonomies have been used to perform hierarchical classification of music genres and color images. In order to generate taxonomies, we adopted the following methodology that shares some similarities with the methodology applied in [16]: Consider a database $C = \{c_1, \ldots, c_m\}$ composed of m classes. Every class c_i contains $|c_i|$ elements $(1 \leq i \leq m)$, each element $s \in c_i$ being represented by one histogram-based descriptor h_s. Therefore, c_i can be observed as a set $H_i = \left(\bigcup_{s \in c_i} h_s\right)$ of histograms. A vector $\overrightarrow{U_i}$ is first associated with each set H_i, the jth component u_{ij} of $\overrightarrow{U_i}$ being the proposed similarity between the sets H_i and H_j as described in Eq. 28.

$$\begin{cases} \overrightarrow{U_i} = [u_{i1}, u_{i2}, \ldots, u_{im}] & \text{where} \\ u_{ij} = \sigma(H_i, H_j) & \text{with} \quad (1 \leq i,j \leq m). \end{cases} \tag{28}$$

Thereafter, the *Agglomerative Hierarchical Clustering* (AHC) algorithm [37] is applied on the vectors $\overrightarrow{U_i}$ $(1 \leq i \leq m)$ to generate a dendrogram from which the taxonomy of the database will be derived. Figure 12 summarizes the proposed methodology for automatic taxonomy generation. The *AHC* algorithm involves the use of a distance between two vectors and a distance between two clusters. The following distances have been selected:

1. Between vectors: L_1 (*Manhattan*), L_2 (*Euclidean*).
2. Between clusters: *Average* and *Centroid* linkages.

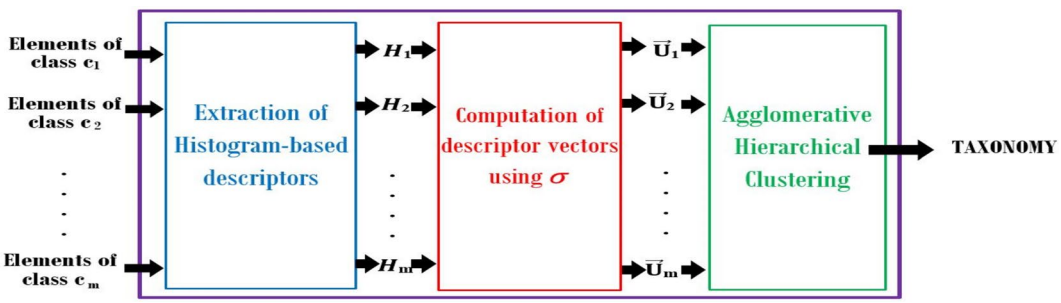

Fig. 12 The proposed methodology for automatic taxonomy generation

4.3.2 Experimental databases

For music genres, the selected databases are *GTZAN* [38] and *GTZAN+* [39]. *GTZAN* is among the most used dataset in music genre recognition, and it contains the 10 following music genres: *blues, classical, country, disco, hiphop, jazz, metal, pop, reggae* and *rock*. Each genre in *GTZAN* is represented by 100 songs, each song having a duration of 30 s. The database *GTZAN+* is an extension of *GTZAN* which is augmented by the five following Afro genres: *bikutsi, makossa, bamileke, salsa* and *zouk*.

For color images, we have selected the same subset of the *Corel* database [40] which was also selected in [17]. The selected subset is composed of the 10 following categories: *card, dinosaur, eagle, ower, gun, postcard, pyramid, ski, sunset* and *tiger*. Each category in *Corel* contains 100 images in the JPEG format, and all the images have the same dimensions which are 126×187 or 187×126.

4.3.3 The selected histogram-based descriptors

For music genres, we have selected the rhythm music descriptors called *Rhythm Histograms* (RHs) proposed by Lidy in [1]. These RHs are among the 401 rhythm music descriptors used in [16]. It is important to mention here that [16] exhibited the actual best hierarchical classification accuracies in the databases *GTZAN* and in *GTZAN+*. Lidy developed an online available MATLAB framework that we used to extract a 60-bins RH for each input song [41].

We have characterized each color image of the *Corel* database by one 64-bins color histogram in the *HSV* color space, following what was done in [17] on this same database.

4.3.4 Specificity of the proposed methodology

There are at least three major differences between [16] and our proposed approach:

1. In [16], the taxonomy generation is exclusively based on many timbre music descriptors, while in this work the taxonomy generation is exclusively based on the RHs which are rhythm music descriptors. In both works, the classification phase is based on the same set of timbre and rhythm music descriptors.
2. A song is represented in [16] as a sequential pattern of music genres. But in this work, a song is represented as a generated Markov chain.
3. In [16], there was no formal algorithm to derive the taxonomy from the dendrogram outputted by *AHC*. But in this paper, the following formal algorithm is proposed to derive the taxonomy: *The taxonomy follows exactly the general shape of the dendrogram, except for the sub-trees of the dendrogram in which each node that is not a leaf has exactly two children including at least one leaf. All the leaves of such sub-trees are merged into a k-ary cluster, where k is the number of leaves in the sub-tree.* When this algorithm is, for example, applied to the dendrogram of Fig. 13a, the taxonomy of Fig. 13b is derived.

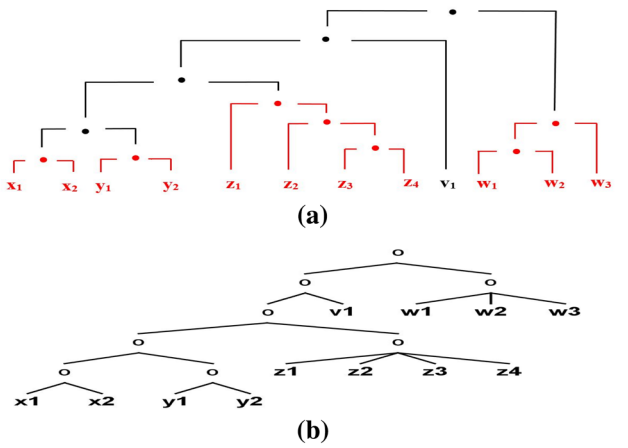

Fig. 13 Principle of taxonomy derivation. The sub-trees that must be merged are colored in red. **a** Input dendrogram. **b** Generated taxonomy

Table 8 The pair of distances that have been used to generate the two taxonomies of *GTZAN*

	Euclidean	Manhattan
Average	T_2	T_1
Centroid	T_1	T_1

4.3.5 Generation of music genre taxonomies

The proposed methodology was applied on *GTZAN* and *GTZAN+* where each genre is considered as a set composed of 100 histograms. The vectors associated with the genres in *GTZAN* and *GTZAN+* were first computed with Eq. 28, and these vectors are, respectively, presented in Table 10a, b. During the computation of these vectors, each HMM training phase took between 17 and 22 min. Finally, the *AHC* algorithm was executed and the resulting dendrograms were used to derive the taxonomies of *GTZAN* and *GTZAN+*. Figure 14a, b shows the two taxonomies T_1 and T_2 that have been derived for *GTZAN*. Table 8 shows the pair of distances that have been used to generate these taxonomies. Only one taxonomy T' was generated for *GTZAN+*, and this taxonomy is presented in Fig. 14c.

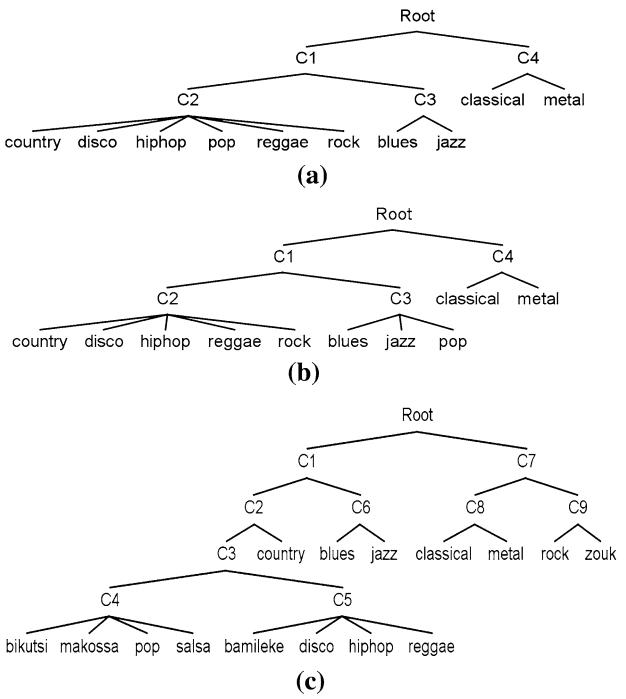

Fig. 14 The taxonomies of *GTZAN* and *GTZAN+*. **a** The taxonomy T_1 generated for *GTZAN*. **b** The taxonomy T_2 generated for *GTZAN*. **c** The taxonomy T' generated for *GTZAN+*

4.3.6 Hierarchical music genre recognition

Similarly to what was done in [16], flat and hierarchical classification experiments have been, respectively, realized in the software WEKA [42, 43] and MEKA [44]. The four following basic classifiers used in [16] with their default settings have also been selected in this work and their WEKA/MEKA corresponding names are in brackets: SVMs with polynomial kernels (*SMO*), multilayers perceptrons (*MLP*), decision trees (*J48*) and KNNs (*IBk*). After a tenfold cross-validation (90% − 10%), the hierarchical classification results in GTZAN and GTZAN+ are, respectively, presented in Table 9a, b. It is with the MLP classifier that the best accuracies 84.9 and 92% were, respectively, obtained in *GTZAN* and *GTZAN+*. We have compared these results with those obtained in [16], and the results are in Table 9c.

As it can be observed in Table 9c, the best accuracy obtained in [16] for *GTZAN* is higher than the best accuracy obtained in this paper for this same database. However, our best accuracy in *GTZAN+* is higher than the best accuracy obtained in [16] for this database. These results enable us to deduce that the genres in *GTZAN* are better characterized by timbre music descriptors, unlike the genres in *GTZAN+* which are better characterized by rhythm music descriptors.

4.3.7 Generation of color images taxonomies

Following the same methodology, we used Eq. 28 to compute a vector for each category in *Corel*. These vectors are presented in Table 10c, and each HMM training phase took between 10 and 15 min during the computation of these vectors. Finally, a unique taxonomy T_c was generated for *Corel*, and this taxonomy is presented in Fig. 15.

Table 9 Hierarchical music genre classification results and comparison with [16]

	Flat	T_1	T_2
(a) In *GTZAN*			
IBk	63.8	74	76.6
SMO	72.3	80.1	82.9
MLP	68.4	81.3	**84.9**
J48	47.3	61.7	66.8
	Flat	T'	
(b) In *GTZAN+*			
IBk	65.4	86.7	
SMO	75.5	91.3	
MLP	41.9	**92**	
J48	47.3	78.3	
	Iloga [16]	This work	
(c)-Comparison between [16] and this work.			
GTZAN	**91.6**	84.9	
GTZAN+	85.2	**92**	

Accuracies are in (%)

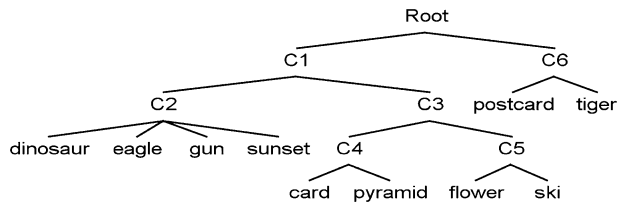

Fig. 15 The taxonomy T_c of the *Corel* database

4.3.8 Hierarchical recognition of color images

In identical experimental conditions, we realized the hierarchical classification of the categories in the *Corel* database. The results presented in Table 11 show that the hierarchical classifiers based on the taxonomy generated by the proposed approach outperformed all the flat classifiers. The best hierarchical classification result is 92.5% obtained once again with the MLP classifier. This performance is very far from the flat classification accuracy of 76.2% exhibited in [17] on this same database and using the same image descriptors. It is important to mention that in [17], the authors did not perform a tenfold cross-validation like in this work. They rather randomly divided the database into two partitions (50–50%), one for the training and the other for testing. They repeated this operation five times, and they took the average accuracy.

4.4 Text document comparison

The last experiment realized in this paper is the use of σ to compare finite sets of text documents. A text document is basically a sequence of symbols. Depending on the language, there exists several tables for matching a symbol with a positive numeric code. The most popular tables for English are the *ASCII* and *EBCDIC* tables [45]. In this paper, the ASCII table is preferred. Many approaches to evaluating the distance between two text documents are presented in [46]. The most widespread principle consists firstly in listing all the words appearing in the two documents. Then for each document, the number of occurrences of each word found in this list is saved into a vector. The final document comparison is performed by computing the distance between these vectors. The goal of this section is to initially transform each document into a set of histograms and then to use σ to compare the documents.

4.4.1 Document transformation

To compare text documents using histograms, each text document t composed of $|t|$ distinct words $\{\omega_1, \ldots, \omega_{|t|}\}$ must initially be transformed into a set H of histograms. If $\omega_j[k]$ refers to the kth symbol in ω_j, then the following transformation is proposed: t is transformed into the set $H_t = \{(a_1.h_1), \ldots, (a_{|t|}.h_{|t|})\}$ of histograms where each a_j $(1 \leq j \leq |t|)$ is the number of occurrences of the word ω_j in t, and each h_j is the histogram verifying $h_j(k) = Ascii(\omega_j[k])$ $(1 \leq k \leq |\omega_j|)$.

4.4.2 Comparison of two documents

In this paper, the comparison is not case-sensitive, but one may decide to proceed otherwise. Consider two text documents t and t' , respectively, transformed into their sets of H_t and $H_{t'}$ of histograms according to the former principle. The similarity rate $\kappa(t, t')$ between t and t' is obtained by computing the similarity rate between H_t and $H_{t'}$ as shown in Eq. 29.

$$\kappa(t, t') = \sigma(H_t, H_{t'}). \tag{29}$$

4.4.3 Comparison of two sets of documents

Consider now two sets $T = \{t_1, \ldots, t_n\}$ and $T' = \{t'_1, \ldots, t'_m\}$ composed of text documents. In order to realize the comparison between T and T', we first transform T into the set $H_T = \left(\bigcup_{i=1}^{n} H_{t_i}\right)$ where every H_{t_i} is the set of histograms associated with the document t_i. The set T' is in a similar way transformed into the set $H_{T'} = \left(\bigcup_{j=1}^{m} H_{t'_j}\right)$ where every $H_{t'_j}$ is set of histograms associated with the document t'_j. Finally, Eq. 29 is applied to obtain the similarity rate between T and T'.

4.4.4 Advantages of κ

The first advantage of κ is that its computation is customizable. In fact, the symbolic table can be *ASCII*, *EBCDIC* or any other valid table for the considered language. Additionally, the comparison can be case-sensitive or not. Another obvious advantage is the fact that the set of histograms associated with a text document t is constructed exclusively with the content of t. Unlike most of the existing approaches that need to construct a vector of words occurrences from the contents of the two documents to be compared. Furthermore, one can compare two single text documents, as well as two sets, each containing several text documents.

4.4.5 Example of comparison of two documents

Table 12 shows details about the contents, the number of words and the sets of histograms of two text documents t and t'. When Eq. 29 is applied on the documents t and t' whose contents are presented in the second line of Table 12, we obtain $\kappa(t, t') = 39.63\%$. Around 15 s were consumed

Table 10 Vectors calculated for *GTZAN*, *GTZAN+* and *Corel*

	Blues	Classical	Country	Disco	Hiphop	Jazz	Metal	Pop	Reggae	rock
(a) Vectors calculated for *GTZAN*										
Blues	100	47.34	63.03	70.40	68.73	75.94	52.36	78.79	69.00	58.73
Classical	47.34	100	60.38	65.53	68.28	51.64	83.27	59.10	63.47	70.84
Country	63.03	60.38	100	85.74	87.17	68.82	65.66	77.59	83.77	76.26
Disco	70.40	65.53	85.74	100	93.28	77.74	71.09	87.36	91.73	82.61
Hiphop	68.73	68.28	87.17	93.28	100	75.37	74.16	84.31	88.98	86.50
Jazz	75.94	51.64	68.82	77.74	75.37	100	56.97	86.00	75.36	65.65
Metal	52.36	83.27	65.66	71.09	74.16	56.97	100	64.25	67.81	77.22
Pop	78.79	59.10	77.59	87.36	84.31	86.00	64.25	100	84.62	74.83
Reggae	69.00	63.47	83.77	91.73	88.98	75.36	67.81	84.62	100	78.70
Rock	58.73	70.84	76.26	82.61	86.50	65.65	77.22	74.83	78.70	100

	bl	cl	co	di	hi	ja	me	po	re	ro	ba	bi	ma	sa	zo
(b) Vectors calculated for *GTZAN+*															
bl	100	47.34	63.03	70.40	68.73	75.94	52.36	78.79	69.00	60.06	71.83	76.40	69.16	77.42	56.72
cl	47.34	100	60.38	65.53	68.28	51.64	83.27	59.10	63.47	70.84	65.59	59.10	57.87	58.39	72.21
co	63.03	60.38	100	85.74	87.17	68.82	65.66	77.59	83.77	77.12	85.84	77.26	76.35	76.09	71.52
di	70.40	65.53	85.74	100	93.28	77.74	71.09	87.36	91.73	82.61	97.27	87.41	85.46	87.04	76.61
hi	68.73	68.28	87.17	93.28	100	75.37	74.16	84.31	88.98	86.50	94.06	84.61	83.00	84.06	80.22
ja	75.94	51.64	68.82	77.74	75.37	100	56.97	86.00	75.36	66.33	78.65	83.10	75.99	83.77	61.63
me	52.36	83.27	65.66	71.09	74.16	56.97	100	64.25	67.81	77.22	71.72	64.48	63.16	64.06	78.85
po	78.79	59.10	77.59	87.36	84.31	86.00	64.25	100	84.62	74.83	89.20	93.62	86.16	94.89	69.32
re	69.00	63.47	83.77	91.73	88.98	75.36	67.81	84.62	100	78.70	92.41	83.28	81.37	82.58	73.42
ro	60.06	70.84	77.12	82.61	86.50	66.33	77.22	74.83	78.70	100	83.31	75.09	73.14	74.41	83.20
ba	71.83	65.59	85.84	97.27	94.06	78.65	71.72	89.20	92.41	83.31	100	89.28	87.40	88.45	77.27
bi	76.40	59.10	77.26	87.41	84.61	83.10	64.48	93.62	83.28	75.09	89.28	100	86.37	92.82	69.39
ma	69.16	57.87	76.35	85.46	83.00	75.99	63.16	86.16	81.37	73.14	87.40	86.37	100	85.79	67.69
sa	77.42	58.39	76.09	87.04	84.06	83.77	64.06	94.89	82.58	74.41	88.45	92.82	85.79	100	68.13
zo	56.72	72.21	71.52	76.61	80.22	61.63	78.85	69.32	73.42	83.20	77.27	69.39	67.69	68.13	100

Table 10 (continued)

	Card	Dinosaur	Eagle	Ower	Gun	Postcard	Pyramid	Ski	Sunset	Tiger
(c) Vectors calculated for *Corel*										
Card	100	68.82	47.76	69.80	59.24	49.65	81.39	67.31	68.62	47.83
Dinosaur	68.82	100	55.12	57.61	68.21	39.80	66.92	55.01	78.31	37.53
Eagle	47.76	55.12	100	40.00	67.40	28.02	46.71	36.84	60.32	28.02
Flower	69.80	57.61	40.00	100	49.32	58.97	79.04	81.43	57.26	56.47
Gun	59.24	68.21	67.40	49.32	100	34.26	56.67	47.82	75.67	33.52
Postcard	49.65	39.80	28.02	58.97	34.26	100	53.83	61.24	41.19	75.20
Pyramid	81.39	66.92	46.71	79.04	56.67	53.83	100	78.48	68.15	53.40
Ski	67.31	55.01	36.84	81.43	47.82	61.24	78.48	100	55.35	59.78
Sunset	68.62	78.31	60.32	57.26	75.67	41.19	68.15	55.35	100	40.40
Tiger	47.83	37.53	28.02	56.47	33.52	75.20	53.40	59.78	40.40	100

Each vector can be read row wise or column wise

Table 11 Hierarchical classification results in *Corel*

	Flat	T_c
IBk	82.1	91.9
SMO	82.3	91.6
MLP	84.0	**92.5**
J48	71.4	87.5

Accuracies are in (%)

by each HMMs training phase during the computation of $\kappa(t, t')$.

5 Conclusion

In this paper, the histograms contained in a finite set H are transformed into generated Markov chains that capture their the bin values and the visual shapes. These generated Markov chains are later used to train a HMM associated with H. Finally, these HMMs are used to perform histogram comparison. The proposed similarity σ between two finite sets H_1 and H_2 of histograms is computed as the similarity rate between their associated HMMs, weighted by a suitable amplitude coefficient. An important asset of σ is that it can be calculated no matter what are the bin sizes of the histograms. Experimented in color image comparison, in the comparison of function curves, in automatic taxonomy generation and in text document comparison, σ exhibited relevant performances which outperformed the existing work in the hierarchical classification of the databases *GTZAN+* and *Corel*.

The following interesting perspectives can be explored in future work:

1. Only the RGB color space was used to compare color images in Sect. 4.1. Whereas if two color images are similar, they might be similar whatever is the selected color space. Future work could focus on the computation of the CSR in other color spaces. It may also be interesting to combine many color spaces during the computation of the CSR.
2. Another possible issue is the use of other distance measures than the *Euclidean distance* to compute the distance between two sets of color images, as well as the distance between the curves of two sets containing 3D functions. Many distance measures listed in Table 1 can be experimented in future work.
3. Future work must also propose a modified computation scheme for the similarity rate between text documents that embed syntactic and semantic information.
4. In Table 5d, the enhanced CSR enabled us to group the 6 images of Fig. 7a–f according to their overall observable color similarities. This result encourages us to use

Table 12 Contents, number of words and sets of histograms of the documents t and t'

Content	Document t			Document t'		
	Yesterday morning, paul went to school after eating because his school is away from home. at the end of the day, paul left school and went home			Paul went to school yesterday after eating in the morning, this is because his school is away from his house. he went back to his house after classes		
Length	$\|t\| = 21$ words			$\|t'\| = 20$ words		
	ω_j	$*$	$h_j \in H_t$	ω'_j	$*$	$h'_j \in H_{t'}$
Set of histo.	Yesterday	1	121 101 115 116 101 114 100 97 121	Paul	1	112 97 117 108
	Morning	1	109 111 114 110 105 110 103	Went	2	238 202 220 232
	Paul	2	224 194 234 216	To	2	232 222
	Went	2	238 202 220 232	School	3	345 297 312 333 333 324
	To	1	116 111	Yesterday	1	121 101 115 116 101 114 100 97 121
	School	3	345 297 312 333 333 324	After	2	194 204 232 202 228
	After	1	97 102 116 101 114	Eating	1	101 97 116 105 110 103
	Eating	1	101 97 116 105 110 103	In	1	105 110
	Because	1	98 101 99 97 117 115 101	The	1	116 104 101
	His	1	104 105 115	Morning	1	109 111 114 110 105 110 103
	Is	1	105 115	This	1	116 104 105 115
	Away	1	97 119 97 121	Is	2	210 230
	From	1	102 114 111 109	Because	1	98 101 99 97 117 115 101
	Home	3	312 333 327 303	His	3	312 315 345
	At	1	97 116	Away	1	97 119 97 121
	The	2	232 208 202	From	1	102 114 111 109
	End	1	101 110 100	House	2	208 222 234 230 202
	Of	1	111 102	He	1	104 101
	Day	1	100 97 121	Back	1	98 97 99 107
	Left	1	108 101 102 116	Classes	1	99 108 97 115 115 101 115
	And	1	97 110 100			

The symbol '*' in this table represents the number of occurrences of each word in the considered document

this enhanced CSR to perform color images clustering in future work.

5. The CSR and the enhanced CSR between the two images presented in Fig. 16a, b (extracted from [35]) are always 100%, whatever is the grid used to compute the enhanced CSR. However, this result is wrong when spatial information are considered. Future work must improve the proposed approach in order to derive new computation schemes that will embed spatial information.
6. As it was experimentally observed, the use of σ in various domains can induce a huge time cost due to the HMMs training phases that are highly time consuming. Therefore, future work should focus on the execution of these HMMs training phases in parallel on separated processors. The time cost can be further reduced by executing parallel versions of the *Bauwm–Welch* algorithm. This can be done on a cluster of computers like in [47] or on a *Field-Programmable Gate Array* (FPGA) chip like in [48].

Table 13 Impact of the maximum number of iterations of the *Baum–Welch* algorithm on the similarities between the 3D functions of Fig. 11a–c and on their corresponding time costs in minutes

W	Z	$\Omega^2_{W,Z}$		$d_\sigma(W,Z)$	$\psi_3(W,Z)$ in %
		x	y		
(a) Similarities for 500 iterations					
$\{f_1\}$	$\{f_2\}$	57.64	30.76	81.17	42.60
$\{f_1\}$	$\{f_3\}$	38.25	46.43	81.75	42.20
$\{f_2\}$	$\{f_3\}$	29.96	29.99	99.03	29.98
(b) Similarities for 50 iterations					
$\{f_1\}$	$\{f_2\}$	55.66	32.98	80.36	43.18
$\{f_1\}$	$\{f_3\}$	40.18	46.42	80.30	43.22
$\{f_2\}$	$\{f_3\}$	31.08	29.99	98.24	30.54
W		W_2^x		W_2^y	
(c) Time for 500 iterations					
$\{f_1\}$		$45'$		$20'$	
$\{f_2\}$		$22'$		$37'$	
$\{f_3\}$		$4'$		$4'$	
(d) Time for 50 iterations					
$\{f_1\}$		$3'30$s		$3'50$s	
$\{f_2\}$		$3'30$s		$4'$	
$\{f_3\}$		$0'27$s		$0'30$s	

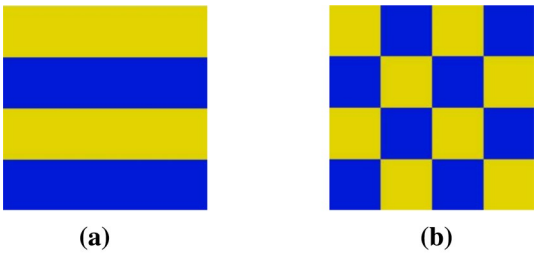

Fig. 16 Images that demonstrate the lack of spatial information during the computation of the CSR. **a** Image 1. **b** Image 2

7. In this paper, the maximum number of iterations of the *Bauwm–Welch* algorithm is set to 500, which is indeed a high value. For example, we have once again compared the three 3D functions f_1, f_2 and f_3 of the example presented in Sect. 4.2.3 with a maximum number of iterations of 50 for the *Baum–Welch* algorithm. The results exhibited in Table 13 prove that the time costs are considerably reduced and the impact on the final similarities is acceptable. An analysis must be realized in future work to study the impact of the reduction of this maximum value on the similarity rate accuracy.

References

1. Thomas L, Rauber A (2005) Evaluation of feature extractors and psycho-acoustic transformations for music genre classification. In: ISMIR, pp. 34–41
2. Swain MJ, Ballard DH (1991) Color indexing. Int J Comput Vis 7(1):11–32
3. Manjunath BS, Ohm J-R, Vasudevan VV, Yamada A (2001) Color and texture descriptors. IEEE Trans Circuits Syst Video Technol 11(6):703–715
4. Tamura H, Mori S, Yamawaki T (1978) Textural features corresponding to visual perception. IEEE Trans Syst Man Cybern 8(6):460–473
5. Hall G (2015) Pearson's correlation coefficient. Url, http://www.hep.ph.ic.ac.uk/~hallg/UG_2015/Pearsons.pdf. Accessed Feb 2017
6. Jurman G, Riccadonna S, Visintainer R, Furlanello C (2009) Canberra distance on ranked lists. In: Proceedings of advances in ranking NIPS 09 workshop, pp 22–27
7. http://stats.stackexchange.com/questions/7400/how-to-assess-the-similarity-of-two-histograms
8. Deselaers T, Keysers D, Ney H (2008) Features for image retrieval: an experimental comparison. Inf Retr 11(2):77–107
9. Kapur JN, Esavan HK (1992) Entropy optimization principles and their applications. In: Entropy and energy dissipation in water resources. Springer, pp 3–20
10. Hafner J, Sawhney HS, Equitz W, Flickner M, Niblack W (1995) Efficient color histogram indexing for quadratic form distance functions. IEEE Trans Pattern Anal Mach Intell 17(7):729–736

11. Pele O, Werman M (2010) The quadratic-chi histogram distance family. In: European conference on computer vision. Springer, pp 749–762
12. Ling H, Okada K (2006) Diffusion distance for histogram comparison. IEEE Comput Soc Conf Comput Vis Pattern Recognit 1:246–253
13. Rubner Y, Tomasi C, Guibas LJ (2000) The earth mover's distance as a metric for image retrieval. Int J Comput Vis 40(2):99–121
14. Rubner Y, Puzicha J, Tomasi C, Buhmann JM (2001) Empirical evaluation of dissimilarity measures for color and texture. Comput Vis Image Underst 84(1):25–43
15. Kedem D, Tyree S, Sha F, Lanckriet GR, Weinberger KQ (2012) Non-linear metric learning. In: Pereira F, Burges CJC, Bottou L, Weinberger KQ (eds) Advances in neural information processing systems 25. Curran Associates, Inc., pp 2573–2581. http://papers.nips.cc/paper/4840-non-linear-metric-learning.pdf
16. Iloga S, Romain O, Tchuenté M (2016) A sequential pattern mining approach to design taxonomies for hierarchical music genre recognition. Pattern Anal Appl. https://doi.org/10.1007/s10044-016-0582-7
17. Li F, Dai Q, Xu W, Er G (2007) Histogram mining based on Markov chain and its application to image categorization. Signal Process Image Commun 22(9):785–696
18. Megshi K, Ishii S (2015) Expanding histogram of colors with gridding to improve tracking accuracy. In: IAPR international conference on machine vision applications (MVA). IEEE, pp 475–479
19. Nikulin MS (2001) Hellinger distance. Encycl Math 78
20. Cha S-H, Srihari SN (2002) On measuring the distance between histograms. Pattern Recognit 35(6):1355–1370
21. Serratosa F, Sanfeliu A (2005) A fast distance between histograms. In: Iberoamerican congress on pattern recognition. Springer, pp. 1027–1035
22. Ionescu RT, Popescu M (2016) Knowledge transfer between computer vision and text mining: similarity-based learning approaches. Adv Comput Vis Pattern Recognit. Springer. ISBN: 973-3-319-30365-9
23. Luo Y, Liu T, Tao D, Xu C (2014) Decomposition-based transfer distance metric learning for image classification. IEEE Trans Image Process 23(9):3789–3801
24. Luo Y, Wen Y, Tao D (2017) Heterogeneous multitask metric learning across multiple domains. IEEE Trans Neural Netw Learn Syst 23(9):3789–3801
25. Rabiner LR (1989) A tutorial on hidden Markov models and selected applications in speech recognition. Proc IEEE 77(2):257–286
26. Falkhausen M, Reininger H, Wolf D (1995) Calculation of distance measures between hidden Markov models. EUROSPEECH
27. Bahlmann C, Burkhardt H (2001) Measuring hmm similarity with the Bayes probability of error and its application to online handwriting recognition. In: Proceedings of the 6th ICDAR. IEEE, pp 406–411
28. Chen L, Man H (2005) Fast schemes for computing similarities between Gaussian HMMs and their applications in texture image classication. EURASIP J. Appl. Signal Process 13:1984–1993
29. Do M (2003) Fast approximation of kullback-leibler distance for dependence trees and Hidden Markov Models. Signal Process Lett 10(4):115–118
30. Silva J, Narayanan S (2008) Upper bound Kullback–Leibler divergence for transient Hidden Markov Models. IEEE Trans Signal Process 56(9):4176–4188
31. Lyngso RB, Pedersen CN, Nielsen H (1999) Metrics and similarity measures for Hidden Markov Models. In: International conference on intelligent systems for molecular biology, pp 178–186
32. Zeng J, Duan J, Wu C (2010) A new distance measure for Hidden Markov Models. Expert Syst Appl 37(2):1550–1555
33. Sahraeian SME, Yoon B-J (2011) A novel low-complexity HMM similarity measure. Signal Process Lett 18(2):87–90
34. Iloga S, Romain O, Lotfi B, Tchuenté M (2014) Musical genres classification using Markov models. In: International conference on audio, language and image processing (ICALIP). IEEE, pp 701–705
35. Schettini R, Ciocca G, Zuffi S (2001) A survey of methods for colour image indexing and retrieval in image databases. In: Color imaging science: exploiting digital media. Wiley, pp. 183–211
36. https://www.math.uri.edu/~bkaskosz/flashmo/graph3d/
37. Shao X, Xu C, Kankanhalli MS (2004) Unsupervised classification of music genre using hidden Markov model. In: IEEE international conference on multimedia and expo (ICME'04), vol 3. IEEE, pp. 2023–2026
38. http://marsyasweb.appspot.com/download/data_sets/
39. http://perso-etis.ensea.fr/sylvain.iloga/GTZAN+/
40. http://www.ci.gxnu.edu.cn/cbir/Corel.zip
41. www.ifs.tuwien.ac.at/mir/muscle/del/audio_extraction_tools.html
42. Witten IH, Frank E (2005) Data mining: practical machine learning tools and techniques. Morgan Kaufmann, Burlington
43. http://weka.sourceforge.net/
44. http://meka.sourceforge.net/
45. http://www.simotime.com/asc2ebc1.htm
46. Huang A (2008) Similarity measures for text document clustering. In: New Zealand computer science research student conference (NZCSRSC), Christchurch, New Zealand, pp 49–56
47. Anikeev M, Makarevich O (2006) Parallel implementation of Baum–Welch algorithm. In: Proceedings of workshop on computer science and information technologies (CSIT'06), vol 1, Karlsruhe, Germany, pp 197–200
48. Espinosa-Manzo A, López-López A, Arias-Estrada MO (2001) Implementing hidden Markov models in a hardware architecture. In: Proceedings international meeting of computer science (ENC'01), vol II, Aguascalientes, Mexico, pp 1007–1016