A New Algorithm to Compute the Distance between Multi-dimensional Histograms

Francesc Serratosa[1], Gerard Sanromà[1] & Alberto Sanfeliu[2]

[1] Universitat Rovira i Virgili, Spain
[2] Universitat Politècnica de Catalunya, Spain
francesc.serratosa@.urv.cat
gerard.sanroma@urv.cat
sanfeliu@iri.upc.edu

Abstract.
The aim of this paper is to present a new algorithm to compute the distance between n-dimensional histograms. There are some domains such as pattern recognition or image retrieval that use the distance between histograms at some step of the classification process. For this reason, some algorithms that find the distance between histograms have been proposed in the literature. Nevertheless, most of this research has been applied on one-dimensional histograms due to the computation of a distance between multi-dimensional histograms is very expensive. In this paper, we present an efficient method to compare multi-dimensional histograms in $O(z^2)$, where z represents the number of bins.

1. Introduction

Finding the distance or similarity between histograms is an important issue in image classification or image retrieval since a histogram represents the frequency of the values of the pixels among the images. For this reason, a number of measures of similarity between histograms have been proposed and used in computer vision and pattern recognition. Moreover, if the position of the pixels is unimportant while considering the distance measure, we can compute the distance between images in an efficient way by computing the distance between their histograms.

Histograms can also be used in structural pattern recognition. For instance, Serratosa defined the Function-Described Graphs [12], which is structure that represents a cluster of Attributed Graphs in which there is a probability density function in each node of the structure described by a histogram. Thus, to compare clusters (that is, to compare Function-Described Graphs), it is needed a distance between histograms to compare each of their nodes. Latter, the same authors defined the Second-Order Random Graphs [13]. This structure represents also a cluster of Attributed Graphs but there is much amount of information since there is a joint probability in each node described by a 2-dimensional histogram. The computational cost of comparing graphs is exponential respect the number of nodes in the worst case. There are some efficient algorithms that obtain sub-optimal distances in polynomial cost respect the number of nodes. For this reason, it is important to reduce the time consuming comparing their nodes.

Most of the distance measures presented in the literature (there is an interesting compilation in [1]) consider the overlap or intersection between two histograms as a function of the distance value but they do not take into account the similarity on the non-overlapping parts of the two histograms. For this reason, Rubner presented in [2] a new definition of the distance measure between n-dimensional histograms that

overcomes this non-overlapping parts problem. It was called Earth Mover's Distance and it is defined as the minimum amount of work that must be performed to transform one histogram into the other one by moving distribution mass.

Often, for specific set measurements, only a small fraction of the *bins* in a histogram contain significant information, that is, most of the *bins* are empty. This is more frequent when the dimensions of the histograms increase. In that cases, the methods that use histograms as fixed-sized structures obtain poor efficiency. In the algorithm depicted by Rubner [2] to find the Earth Mover's Distance the empty-bins where not explicitly considered. They used the simplex algorithm [3] to compute the distance measure and the method presented in [4] to search a good initialisation. The computational cost of the simplex iteration is $O(z'^2)$, where z' is the number of non-empty bins. The main drawback of this method is that the number of iterations is not bounded. Moreover, the initialisation cost is $O(z'^3)$.

To reduce the computational cost, Cha presented in [1] three algorithms to obtain the Earth Mover's Distance between one-dimensional histograms when the type of measurements where *nominal*, *ordinal* and *modulo* in $O(z)$, $O(z)$ and $O(z^2)$ respectively, being z the number of levels or bins.

Finally, Serratosa reduced more the computational cost of comparing one-dimensional histograms in [5]. They presented three new algorithms to compute the Earth Mover's Distance between one-dimensional histograms when the type of measurements where *nominal*, *ordinal* and *modulo*. The computational cost were reduced to $O(z')$, $O(z')$ and $O(z'^2)$ respectively, being z' the number of non-empty bins.

It was presented in [6] an algorithm to compute the distance between histograms that the input was a built histogram (obtained from the target image) and another image. Then, it was not necessary to build the histogram of the image of the database to compute the distance between histograms.

Really few have been done to compare n-dimensional histograms except in [2]. The main drawback of the method presented in [2] is the computational cost. The following papers, make use of colour histograms, although the distance between them is not the main object of the work. One of the earliest papers is [7]. In that paper, the intersection of a pair of colour histograms (three dimensional histogram) was used to obtain a similarity measure between images. More recently, some kernel functions were defined in [8] based on the RGB and HSV histograms. In that paper, the number of bins per each dimension of the histograms had to be reduced to 16 due to run time an space requirements. And in [9], a support vector machine was used to classify images based on colour histograms. The bins of their histograms were not fixed-structures but they were variable depending on the density of the pixels. Finally, in [10], a tree structure was defined for image retrieval. In each node of the tree, only the average of the histograms were stored.

In this paper, we present an efficient algorithm to compute the distance between n-dimensional histograms with a computational cost of $O(z^2)$. Our algorithm does not depend on the type of measurements (*nominal*, *ordinal* or *modulo*). In the next section, we define the histograms and types of values. In section 3, we give the definitions of distances between histograms and between sets and in section 4 we show the algorithm to compute the distance between histograms. In section 5, we show a separability-class function based on the histogram distance. In sections 6 and 7 we show the experimental validation of our algorithm and the conclusions.

2. Sets and Histograms

Let x be a measurement which can have one of z values contained in the set $X=\{x_1,...x_z\}$. Each value can be represented in a T-dimensional vector as $x_i=(x_i^1, x_i^2,\ldots,x_i^T)$. Consider a set of n elements whose measurements of the value of x are $A=\{a_1,...a_n\}$ where $a_t \in X$ being $a_t=(a_t^1, a_t^2,\ldots,a_t^T)$.

The histogram of the set A along measurement x is $H(x,A)$ which is an ordered list consisting of the number of occurrences of the discrete values of x among the a_t. As we are interested only in comparing the histograms and sets of the same measurement x, $H(A)$ will be used instead of $H(x,A)$ without loss of generality. If $H_i(A)$, $1 \le i \le z$, denotes the number of elements of A that have value x_i, then $H(A)=[H_1(A), \ldots, H_z(A)]$ where

$$H_i(A) = \sum_{t=1}^{n} C_{it}^{A} \tag{1}$$

and the individual costs are defined as

$$C_{i,t}^{A} = \begin{cases} 1 & \text{if } a_t = x_i \\ 0 & \text{otherwise} \end{cases} \tag{2}$$

The elements $H_i(A)$ are usually called *bins* of the histogram. Note that z is the number of bins of the histogram. In a T-dimensional histogram with m values per each dimension, the number of bins is $z=m^T$. Therefore, $1 \le i \le m^T$.

In this paper, the sets are images. For this reason and for the rest of the paper, the bins of the histograms represent the null or natural numbers, $a_t^i \in \{0 , N\}$.

The distance between histograms presented in this paper is used as a fast method for comparing images and image retrieval. The most used colour representations are base on the R,G,B or H,S,I descriptors. The hue parameter (H) is a modulo-type measurement (measurement values are ordered but form a ring due to the arithmetic modulo operation) and the other parameters are ordinal-type measurements.

Corresponding to these types of measurements mentioned before, we define a measure of difference between two measurement levels $a=(a^1, a^2,\ldots,a^T) \in X$ and $b=(b^1, b^2,\ldots,b^T) \in X$ as follows:

$$d(a,b) = \sqrt{\sum_{j=1}^{T} S^2} \qquad \text{where} \tag{3}$$

$$S = \begin{cases} m - \left|a^j - b^j\right| & \text{if } a^j - b^j \le m/2 \text{ and } a^j, b^j \in \text{Modulo type} \\ \left|a^j - b^j\right| & \text{otherwise} \end{cases} \tag{4}$$

This measure satisfy the following necessary properties of a metric. Since they are straightforward facts, we omit the proofs. The proof of the triangle inequality for the modulo distance is depicted in [1] for the one-dimensional case ($T=1$).

3. Distance definitions

In this section we present the distance between sets $D(A,B)$ and the distance between their histograms $D(H(A),H(B))$. It was demonstrated in [11] that both satisfy the necessary properties of a metric and that the distance values are the same, $D(A,B) = D(H(A),H(B))$. This is an important result since we present an algorithm that obtains a good approximation of $D(H(A),H(B))$ with a quadratic computational cost respect the number of *bins* of the histogram z. Moreover, in most of the applications, z is much

smaller than *n*. Another advantage is that the time consuming of the comparison is constant and does not depend on each set.
Given two sets of *n* elements, *A* and *B*, the distance measure is considered as the problem of finding the minimum difference of pair assignments between both sets. That is, to determine the best one-to-one assignment *f* (bijective function) between the sets such that the sum of all the differences between two individual elements in a pair $a_i \in A$ and $b_{f(i)} \in B$ is minimised. See [11] for more information.

$$D(A,B) = \min_{\forall f A \to B} \left(\sum_{t=1}^{n} d\left(a_t, b_{f(t)}\right) \right) \tag{5}$$

The distance between histograms that our algorithm computes was presented in [11]. It is a generalisation of the Earth Mover's Distance presented in [2]. Intuitively, given two T-dimensional histograms, one can be seen as a mass of earth properly spread in space, the other as a collection of holes in that same space. Then, the distance measure is the least amount of work needed to fill the holes with earth. Here, a unit of work corresponds to transporting a unit of earth by a unit of ground distance.
More formally, given two histograms *H(A)* and *H(B)*, where measurements can have one of *z* values contained in the set $X=\{x_1,...x_z\}$, the distance between the histograms *D(H(A),H(B))* is defined as follows,

$$D(H(A), H(B)) = \min_{\forall f A \to B} \left(\sum_{i,j=1}^{z} d\left(x_i, x_j\right) g_f(i,j) \right) \tag{6}$$

The flow between the *bins* of both histograms is represented by $g_f(i,j)$, that is, the mass of earth that is moved as one unit from the *bin i* to the *bin j*. The product $d(x_i,x_j)g_f(i,j)$ represents the work needed to transport this mass of earth. Notice that $z = m^T$, being *m* the number of bins per each dimension and *T* the number of dimensions.
In [11], we determine the flow between *bins* $g_f(i,j)$, as a function of the one-to-one assignment *f* between the sets *A* and *B* used to compute the distance *D(A,B)* as follows,

$$g_f(i,j) = \sum_{t=1}^{n} C_{i,t}^{A} C_{j,f(t)}^{B} \quad 1 \le i, j \le z \tag{7}$$

were the costs *C* are given in (2).
It was demonstrated in [11] that with this new definition, we obtain two advantages; First, there is a relation between distances *D(A,B)* and *D(H(A),H(B))* through their definition. Second, the constraints arbitrarily imposed to the flow between *bins* in [2], were converted in deducted properties that make possible to naturally match the distance between histograms to the transportation problem.

4. Algorithm

In this section, we depict an efficient algorithm used to compute the distance between histograms based on a solution to the well-known transportation problem [3]. Suppose that several suppliers, each with a given amount of goods, are required to supply several consumers, each with a given limited capacity. For each pair of suppliers and consumers, the cost of transporting a single unit of goods is given. The transportation problem is then to find a least-expensive flow of goods from the suppliers to the consumers that satisfies the consumer's demand. Our distance between histograms can be naturally cast as a transportation problem by defining one histogram as the

supplier and the other one as the consumer. The cost of transporting a single unit of goods is set to the distance between the *bin* of one histogram and the *bin* of the other one, $d(x_i,x_j)$. Intuitively, the solution of the transportation problem, $g_f(i,j)$, is then the minimum amount of "work" required to transform one histogram to the other one subjected to the constraints defined by the properties of the flow $g_f(i,j)$ (see [11]).
The computational cost of the transportation problem is exponential, respect the number of suppliers and consumers, that is, the number of bins of the histograms, *z*. Fortunately, efficient algorithms are available. One of the most common solutions is the simplex algorithm [3], which is an iterative method that the cost of one simplex iteration is $O(z^2)$. The main drawback is that the number of iterations is not bounded and that this method needs a good initial solution. The Russell method [4] is the most common method used to find the first solution with a computational cost of $O(z^3)$.
In this paper, we present an efficient and not iterative algorithm (figure 1) with a computational cost of $O(z^2)$.
Given a pair of bins from both histograms, *i* and *j*, our algorithm finds the amount of goods that can be transported, $g_f(i,j)$, and computes the cost of this transportation, $g_f(i,j)*d(x_i,x_j)$. The algorithm finishes when all the goods have been transported, that is, all the elements of the sets, *n*, have been considered. In each iteration, a pair of bins is selected by the function *next*, in a given order.

```
Algorithm Histogram-Distance (H(A),H(B))
i,j = first()
while n > 0 // n: the number of elements of both sets
       gf(i,j) = min (Hi(A) , Hj(B))
       Hi(A) = Hi(A) - gf(i,j)
       Hj(B) = Hj(B) - gf(i,j)
       n = n - gf(i,j)
       D = D + gf(i,j) * d(xi,xj)
       i,j = next (i ,j)
Return D  //distance between histograms
```

Figure 1. Algorithm that computes the distance between n-dimensional histograms.

4.1. The *next* function

Given a pair *i,j* (*i* and *j* are a supplier and a consumer, respectively), the *first* and *next* function returns the first and next pairs of supplier and consumer to be explored, respectively. The first pair of supplier-consumer and the order generated by the *next* function only depends on the dimensionality of histogram and the number of bins but not on the values of the histograms, for this reason, *first* and *next* can be computed a priori.
The order of the pairs *i,j* is set by decrementing an energy function *E* as follows,

$$i',j' = next\ (i,j) \qquad iff \qquad E(i',j') \le E(i,j) \tag{8}$$

where *E* is defined as,

$$E(i,j) = Path\_Deviation_j(i) + Path\_Deviation_i(j) \tag{9}$$

The $Path\_Deviation_j(i)$ is the difference between the maximum cost from the bin *i* to any bin of the histogram and the real cost from this bin to the bin *j*,

$$Path\_Deviation_j(i) = \max\_dist(x_i) - d(x_i, x_j) \tag{10}$$

It represents the worst case that the good can be sent (supplier) or received (consumer) respect the best case. Note that several pairs *i,j* can obtain the same energy value. In those cases, the order between them is set arbitrarily.

4.2. Computational cost

Each step of the loop of the algorithm has a constant computational cost. The *next* function is implement as an array that for each pair *i,j*, returns the next pair *i',j'*. For this reason, the worst computational cost of our algorithm only depends on the number of iterations. The algorithm finishes when all the goods, *n*, have been transported and so, the worst case would be in the case that this is achieved at the last transportation from *i,j*, to *i',j'*. The number of possible transportations is z^2.

5. Experimental Validation

To verify the performance of the proposed method, we have conducted an experiment on the WANG database. The WANG database is a subset of the Corel database of 1000 images, which were selected manually to form 10 classes of 100 images each. The images are subdivided into 10 classes (e.g. Africa, beach, ruins, food) such that it can be assumed that a user wants to find the other images from a class if the query is from one of these 10 classes.

Our test database was composed by 200 images from the WANG database (20 images for each class). In Figure 2 we can see one representative image of each class. The query set was composed by 20 images (2 images for each class) and the database set was composed by the other 180 images (18 images for each class). We decide that the query image belongs to a class using the 5 nearest neighbours criteria.

Figure 2. Ten images of WANG database. One of each class.

Given the depth of the pixels, the number of colours of the image, *z*, and also the computational cost of our algorithm is obtained. Besides, we can deduct the number of bins per dimension, *m*, depending on the dimensionality of the histogram. Table 1 shows in the last three columns the possible combinations of dimensionality and number of bins per dimension. For instance, a 1D-histogram with 64 bins has the same computational cost, 4096, than a 2D-histogram with 8 bins per dimension and a 3D-histogram with 4 bins per dimension. The empty cells in the table are the ones that *m* is not a natural number.

depth of pixels	z (#colours)	Cost $O(z^2)$	m (1D) $(z=m^1)$	m (2D) $(z=m^2)$	m (3D) $(z=m^3)$
1	2	4	2		
2	4	16	4	2	
3	8	64	8		2
4	16	256	16	4	
5	32	1024	32		
6	64	4096	64	8	4
7	128	16384	128		
8	256	65536	256	16	
9	512	262144	512		8
10	1024	1048576	1024	32	
11	2048	4194304	2048		
12	4096	16777216	4096	64	16

Table 1. Relation between the number of bins per dimension, *m*, and the number of colours, *z*, and cost, $O(z^2)$ in 1D, 2D and 3D histograms.

Figure 3 shows the recognition ratio respect the number of colours in 6 different cases. In the first two cases, we have considered only the hue and the luminance. In the other two, we have considered the hue and saturation and also hue and luminance. In the last two cases we have considered the red, green and blue channels and also the hue, saturation and luminance channels. The values in figure 3 that correspond to empty cells in table 1 have been interpolated.

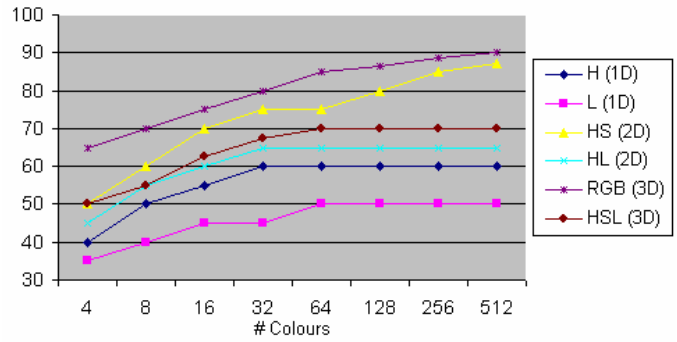

Figure 3. Recognition ratio respect the number of colours.

As we could imagine, when the number of colours increases (that is, the depth of the pixels), also increases the recognition ratio. But also, given a number of colours, it is worth to increase the dimensionality, since the recognition increases with the same computational cost. We can see also, that in this experiments, the Luminance has poor information, since H(1D) has better results than L(1D) and HS(2D) has better results than HSL(3D) or HL(2D).

6. Conclusions and future work

We have presented a new distance between multi-dimensional histograms and an efficient algorithm to compute this distance. Our method is useful for comparing multi-dimensional histograms of any type of measurements. The theoretical

computational cost is $O(z^2)$, being z the number of bins. From the application point of view, we have seen that it is worth increasing the number of dimensions and reducing the number of bins per each dimension than reducing the number of dimensions in despite of increasing the bins per dimension, with a similar computational cost.
Although the computational cost of $O(z^2)$ can be considered low, in a real application, the run time can be too long when the number of images to be compared and the number of colours per image is high. For this reason, we are thinking about applying the *signature* structure presented in [6] due to the fact that in this structure, the empty bins are not explicitly considered. We will have the advantage that the number of iterations would be reduced (in the cases that the number of colours is high, that is, the histograms are sparse). But we will have the drawback that the computational cost of the function *next* would not be constant. We leave as a future work the implementation of this new method. From a theoretical point of view, we will propose the new algorithm. And from the practical point of view, we have to show if the reduction of the number of iterations compensates the increase of the computational cost of the function *next*.

7. References

1. S.-H. Cha, S. N. Srihari, "On measuring the distance between histograms" *Pattern Recognition* 35, pp: 1355–1370, 2002.
2. Y. Rubner, C. Tomasi, and L. J. Guibas, "A Metric for Distributions with Applications to Image Databases" *International Journal of Computer Vision* 40 (2), pp: 99-121, 2000.
3. *Numerical Recipes in C: The Art of Scientific Computing*, ISBN 0-521-43108-5.
4. E. J. Russell. "Extension of Dantzig's algorithm to finding an initial near-optimal basis for the transportation problem",*Operations Research*(17),pp:187-191, 1969.
5. F. Serratosa & A. Sanfeliu, "Signatures versus Histograms: Definitions, Distances and Algorithms", *Pattern Recognition* (39), Issue 5, pp. 921-934, 2006.
6. F.-D. Jou, K.-Ch. Fan, Y.-L. Chang, "Efficient matching of large-size histograms", *Pattern Recognition Letters* 25, pp: 277–286, 2004.
7. M.J. Swain and D.H.Ballard, "Indexing via colour histograms", *Int. J. Computer Vision*, (7), pp: 11-32, 1991.
8. M. Kolesnik & A. Fexa, "Multi-dimensional colour Histograms for Segmentation of Wounds in Images",ICIAR 2005, LNCS 3656, pp:1014-1022, 2005.
9. O. Chapelle, P.Haffner and V.N. Vapnik, "Support Vector Machines for Histogram-Based Image Classification", *IEEE Trans. On Neural Net.* (10), 1999.
10. Y. Gong, C. H. Chuan & G. Xiaoyi, "Image indexing and retrieval based on colour histograms". *Multimedia Tools and Applications*, (2) issue 2, pp:133 – 156, 1996.
11. F.Serratosa & G.Sanromà, "An Efficient Distance between Multi-dimensional Histograms for Comparing images", *Proc. Syntactic and Structural Pattern Recognition, SSPR'2006, LNCS* , 2006.
12. F. Serratosa, R. Alquézar & A. Sanfeliu, "Function-Described Graphs for modeling objects represented by attributed graphs", *Pattern Recognition*, 36 (3), pp. 781-798, 2003.
13. A. Sanfeliu, F. Serratosa & R. Alquézar, "Second-Order Random Graphs for modeling sets of Attributed Graphs and their application to object learning and recognition", *International Journal of Pattern Recognition and Artificial Intelligence*, Vol. 18, (3), pp: 375-396, 2004.