Rank-Mixer and Rank-Booster: Improving the Effectiveness of Retrieval Methods*

Sebastian Kreft and Benjamin Bustos

PRISMA Research Group
Department of Computer Science, University of Chile
{skreft,bebustos}@dcc.uchile.cl

Abstract. In this work, we present two algorithms to improve the effectiveness of multimedia retrieval. One, as earlier approaches, uses several retrieval methods to improve the result, and the other uses one single method to achieve higher effectiveness. One of the advantages of the proposed algorithms is that they can be computed efficiently in top of existing indexes. Our experimental evaluation over 3D object datasets shows that the proposed techniques outperforms the multimetric approach and previously existing rank fusion methods.

Keywords: Multimedia databases, effectiveness, boosting.

1 Introduction

In the last years, we have experienced a phenomenon of multimedia information explosion, where the volume of produced digital data increases exponentially in time. This exponential growth is caused by many factors, like more powerful computing resources, high-speed internet, and the diffusion of the information society all over the world. Additionally, an enormous production of data is attributed to the quick dissemination of cheap devices for capturing multimedia data like audio, video, and photography. Thus, it has become essential to develop effective methods to search and browse large multimedia repositories.

The *content-based retrieval* (CBR) of multimedia data (or of other semantically unstructured-type data) is a widely used approach to search in multimedia collections. CBR performs the retrieval of relevant multimedia data according to the actual content of the multimedia objects, rather than considering an external description (e.g., annotations). Instead of text-based query, the database is queried by an example object to which the desired database objects should be *similar*. This is known as the *query-by-example* retrieval scheme.

Usually, the similarity measure used to compare two multimedia objects is modeled as a metric distance (in the mathematical meaning), which is known as the *metric space approach* [14]. This is because the metric axioms have allowed researchers to design efficient (fast) access methods employed in the similarity search. With this approach, the search can be performed in an efficient way.

* Partially funded by Conicyt (Chile), through the Master Scholarship (first author).

D. Ünay, Z. Çataltepe, and S. Aksoy (Eds.): ICPR 2010, LNCS 6388, pp. 119–128, 2010.

However, depending on the particular application domain, the similarity measure may not model 100% correctly the human notion of similarity.

The *effectiveness* of a multimedia research system is related with the quality of the answer returned by the similarity query. In the metric approach, given a distance function, a similarity query corresponds to a search for close objects in some topological space. An effective distance function should treat two similar objects, according to the human concept of similarity, as two close points in the corresponding space. Indeed, the effectiveness of a similarity search system measures its ability to retrieve relevant objects while at the same time holding back non-relevant ones. Improving the effectiveness of a similarity search system is at least as important as improving its efficiency, because effectiveness is directly related to the quality of the answers that the search system returns.

In this paper we present two novel algorithms that improves the effectiveness of similarity measures. The first method, *Rank-Mixer*, merges several results obtained with different similarity measures, and the second method, *Rank-Booster*, improves the quality of the answer obtained using just one similarity measure. Both methods only use the information given by the similarity measure, and they do not rely on training databases like other approaches based on off-line supervised learning. We show how to use these methods over existing index structures, thus the efficiency of the search is not affected. To evaluate the performance of our proposed algorithms, we made an extensive experimental evaluation in a standard reference collection for 3D model retrieval and used the data of the ImageCLEF@ICPR fusion task, showing that Rank-Booster and Rank-Mixer are able to improve the effectiveness of the best available 3D model similarity measures and the best textual methods of ImageCLEF.

2 Related Work

2.1 Similarity Queries

The most common similarity query is the nearest neighbors or k-NN, which returns the k most similar objects of the database with respect to a query object q not necessarily present in the database.

The (dis)similarity is defined as a function that takes two multimedia objects as input and returns a positive value. The value 0 means that the objects are equal. Typically, to compute the (dis)similarity between multimedia objects the metric approach is used. In this approach, the dissimilarity is a distance for which the triangle inequality holds. This is done generally by computing for every object a feature vector (FV) that represents the properties of that object.

2.2 Metric Combination

It has been shown [3,4] that a query dependent combination of metric spaces yields to higher effectiveness of the similarity search. One way to combine several metrics is by mean of multi-metric spaces, where the (dis)similarity function is computed as a linear combination of some selected metrics.

Definition 1. *Multi-metric Space*
Let $\mathcal{X} = \{(\mathbb{X}_i, \delta_i),\ 1 \leq i \leq n\}$ *a set of metric spaces, the corresponding* Multi-metric space *is defined as the pair* $(\prod_{i=1}^{n} \mathbb{X}_i, \Delta_{\mathbb{W}})$*, where* $\Delta_{\mathbb{W}}$ *is a linear multi-metric, which means*

$$\Delta_{\mathbb{W}}(x, y) = \sum_{i=1}^{n} w_i \delta_i(x_i, y_i), \qquad (1)$$

In the above definition, the vector of weights $\mathbb{W} = \langle w_i \rangle$ is not fixed, and is a parameter of Δ. When $\forall i\, w_i \in [0, 1] \wedge \exists i\, w_i > 0$, $\Delta_{\mathbb{W}}$ is also a metric.

2.3 Rank Fusion

In the area of information retrieval and pattern recognition, there are several methods that given different ranks of objects improve the effectiveness of the result by combining them. Here we present some of them, for a more detailed survey on these methods see Suen and Lam [12]. Most of these methods give each element a score and then rank them according to the assigned score.

Borda Count [9], originally developed for voting systems, has been widely used in information retrieval. This method gives each element the score $\sum_{r \in \mathcal{R}} r(d)$.

Reciprocal Rank [6] gives each element the score $\sum_{r \in \mathcal{R}} \frac{1}{k + r(d)}$.

Logistic Regression Method [9] solves the problem of Borda Count that does not take into account the quality of the different ranks. The score assigned by this method is $\sum_{r \in \mathcal{R}} w_r r(d)$ where the weights w_r are computed as a logistic regression. This method is similar to the idea of entropy-impurity [3].

Med-Rank [7] is an aggregation method intended for vector spaces. However, it can also be applied to combine different ranks. In this method, each element gets a score equal to the index i, such that it appears at least in $f_{min}|\mathcal{R}|$ different ranks up to position i. Mathematically, the score is $\min\{i \in \{1, \ldots, n\} / |\{r(d)/r(d) \leq i\}| > f_{min}\,|\mathcal{R}|\}$, where f_{min} is a parameter, usually taken as $f_{min} = 0.5$.

3 Improving Effectiveness of Retrieval Methods

3.1 Rank-Mixer

This algorithm combines the answer of different multimedia retrieval methods (not necessarily metrics) and produces a new improved answer. The idea behind this algorithm is that if an object is reported to be similar to the query object by several retrieval methods, then the object should be a relevant one. We give a *score* to the objects according to their position in the rankings. Then, we rank all the objects according to the total score they got.

For each retrieval method, we compute the k-NN. Then we apply a function f^+ to the ranking of each object to assign a *score* to the objects. As we want to give higher scores to the first objects in the ranking, f^+ must be a decreasing function. On the other hand, as we do not have the complete ranking, we need to assign an implicit value of 0 to the unseen objects, thus f^+ must be a positive

```
function mixer(q, 𝕌, ℳ, f⁺, k)
  for each m ∈ ℳ do
    rank ← kNN(q, 𝕌, k)
    for each o ∈ rank do
      mrank[o] ← mrank[o] + f⁺(pos(o, rank))
  Sort descending mrank
  return mrank[1:k]
```

```
function booster(q, 𝕌, m, k, f⁺, k_b)
  rank ← kNN(q, max{k, k_b}, 𝕌)
  for each o ∈ rank do
    trank ← kNN(o, k_b, 𝕌)
    for each p ∈ trank do
      brank[p] ← brank[o] + f⁺(pos(p, trank), pos(o, rank))
  Sort descending brank
  brank ← selectElements(brank)
  return [brank,rank-brank][1:k]
```

Fig. 1. Rank-Mixer and Rank-Booster algorithm

function. Adding a positive constant to f^+ we are able to control how much we "punish" elements not present in all ranks. Then, we add all the scores obtained by the objects in each ranking and rank them according to the final scores. This method is in fact a generalization of both Borda Count and Reciprocal-Rank with a score function $\sum_{r\in\mathcal{R}} f^+(r(d))$. The outline of the algorithm is presented in Fig. 1 (left).

Efficiency. The time needed to perform the query is the time of performing a k-NN query for each retrieval method. These queries can be answered efficiently by using some indexing techniques [2,5].

We could weight each retrieval method, similar to the multi-metric approach, either statically or dynamically at query time. The weighed version of Rank-Mixer has the advantage that the running time does not depend on the weights, opposed to what happens in multi-metric spaces [4].

3.2 Rank-Booster

This algorithm uses a single retrieval method to improve the effectiveness of the answer. This algorithm relies on the fact that good retrieval methods have good results for the first elements. For example, in our experimental evaluation, the nearest-neighbor (NN) is computed correctly with 50-80% and when retrieved the same number of relevant objects (R-Precision) the 50-60% of them are relevant.

In this algorithm, we compute the k-NN for the given query. Then for the first k_b (a parameter of the algorithm) elements of the ranking we perform a k_b-NN query and use a similar strategy as the one used for the mixer method to combine these rankings. The difference between this algorithm and Rank-Mixer, is that we score the objects according to two values. These are the position of each object in the ranking and the position of the object that generated the ranking. Finally, as some objects could get a low score, meaning that they are not good enough, we keep just the first elements of the generated ranking. The outline of this algorithm is presented in Fig. 1 (right).

The function f^+, just as in the fixed Rank-Mixer algorithm, must be positive and decreasing in each coordinate. One variation of the algorithm presented above is to always keep the NN of the original answer.

Efficiency. This algorithm needs no special index to work. It can be built over any existing indexing method. We only need to store the k-NN for each object of the database, thus requiring $(k-1)\,|\mathbb{U}|$ space (the NN of an object of the database is the object itself, thus we do not need to store it). This space may seem a lot, but in fact is lower than the space used by most FVs. For example, as we will show in Section 4.1, for 3D objects $k_b \leq 15$ gives the the best results. And since the dimensionality of the FVs for 3D objects ranges from 30 to over than 500, the space needed would be around 2%–50% of the space needed by a FV. Besides, this information can be dynamically built at query time, thus requiring less space in practice.

4 Experimental Evaluation

Before using our method in the ImageCLEF@ICPR fusion task, we tested our algorithms with two different 3D models datasets: the first one is the dataset of the SHREC 2009 *"Generic retrieval on new benchmark"* track [1], which comes from the NIST generic shape benchmark [8]. The other dataset is the *test* collection from the Princeton Shape Benchmark (PSB) [11][1].

The SHREC dataset is composed of 720 models and 80 query objects. Both models and queries are classified into 40 different classes, each one having exactly 20 objects (18 in the database and 2 queries). The *test* collection of PSB has 907 objects classified into 92 classes. The classes have between 4 and 50 elements. As the PSB does not provide queries for the dataset, we chose the rounded 10% of each class as queries. Thus the *test* collection now have respectively 810 objects in the database and 97 queries.

As retrieval methods we used different FVs with me metric L_1. One of them, the DSR [13] is itself an optimized metric combination, so we are comparing against it in our tests.

4.1 Experimental Results

Rank-Mixer. The first test we performed was intended to evaluate which function performs better for the mixer method. We considered functions of the following forms: $-\log(x)$, $-x^\alpha$, $1/x^\alpha$. Table 1 shows the complete result and Fig. 2 shows the effectiveness of the Rank-Mixer for some of the functions. The table and the graph show that the $f^+(x) = -\log(x)$ gives the best results and clearly outperforms the Borda Count ($f(x) = -x$) and the Reciprocal Rank ($f(x) = 1/x$ or $f(x) = 1/(x + 60)$). In the following tests we will use $f(x) = -\log(x)$ and we will call this method log-rank. The results also show that the log-rank method outperforms MedRank.

[1] We actually tested in both PSB train and test; but we omitted some results because of lack of space.

Table 1. Effectiveness of Rank-Mixer for different functions

Function	NN	1T	2T	E	DCG
$-x^{0.1}$	0.850	0.505	0.637	0.442	0.787
$-\log(x)$	0.863	0.504	0.640	0.443	0.788
$1/x^{0.1}$	0.863	0.503	0.637	0.444	0.788
$-x^{0.25}$	0.850	0.502	0.629	0.440	0.783
$-x^{0.2}$	0.850	0.502	0.631	0.440	0.784
$-x^{0.333}$	0.850	0.496	0.627	0.438	0.782
$1/x^{0.5}$	0.838	0.495	0.628	0.438	0.784

Function	NN	1T	2T	E	DCG
$-x^{0.5}$	0.850	0.493	0.622	0.430	0.781
$-x$	0.850	0.482	0.603	0.418	0.773
$-x^2$	0.838	0.469	0.585	0.409	0.763
$1/x^2$	0.762	0.113	0.137	0.095	0.506
$1/x$	0.762	0.113	0.137	0.095	0.506
$1/(x+60)$	0.025	0.025	0.050	0.032	0.324
MedRank	0.850	0.484	0.622	0.433	0.777

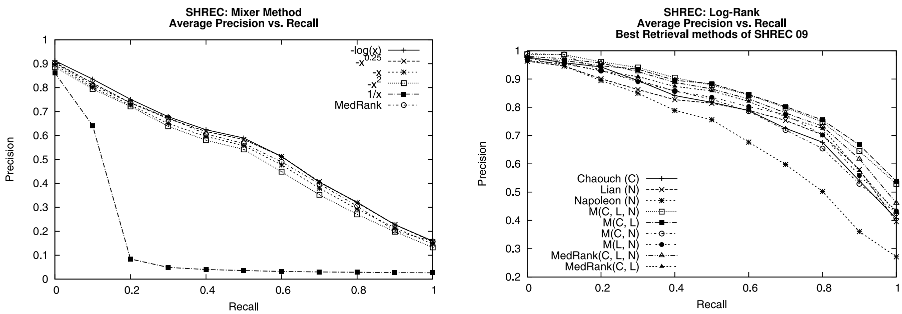

Fig. 2. Left: Effectiveness of Rank-Mixer for different functions, right: Effectiveness of best SHREC descriptors

Table 2. Left: Effectiveness of log-rank on SHREC for best 3D retrieval methods. Right: Effectiveness of Rank-Booster on SHREC.

Method	NN	1T	2T	E	DCG
Chaouch(C)	0.963	0.730	0.848	0.602	0.917
Lian(L)	0.925	0.724	0.844	0.595	0.904
Napoleon(N)	0.950	0.639	0.771	0.541	0.882
M(C, L, N)	0.975	0.781	0.895	0.638	0.945
M(C, L)	0.950	0.788	0.906	0.642	0.938
M(C, N)	0.975	0.728	0.842	0.595	0.924
M(L, N)	0.938	0.733	0.865	0.612	0.922
MedRank(C,L,N)	0.950	0.774	0.891	0.632	0.936
MedRank(C,L)	0.925	0.751	0.865	0.611	0.922

Method	NN	1T	2T	E	DCG
SIL	0.775	0.435	0.582	0.404	0.744
B(SIL)	0.775	0.460	0.590	0.409	0.745
DBD	0.825	0.417	0.541	0.377	0.735
B(DBD)	0.825	0.453	0.589	0.408	0.739
RSH	0.750	0.384	0.504	0.347	0.705
B(RSH)	0.750	0.412	0.502	0.350	0.698
DSR	0.850	0.546	0.691	0.479	0.819
B(DSR)	0.850	0.592	0.717	0.500	0.821

Figure 3 shows that log-rank is similar to the multimetric approach, it also shows that the proposed method outperforms the DSR and the multimetric approach. Also, in the right figure we present two upper bounds that can be obtained using the log-rank method, the first is obtained using the best static combination of retrieval methods and the second one is obtained using the best possible dynamic combination. Figure 2 compares log-rank method with the best retrieval methods of SHREC 09 [1], these are Aligned Multi-View Depth Line, Composite Shape Descriptor and Multi-scale Contour Representation. The results are detailed in Table 2. The results shows that the log-rank outperforms MedRank and that it increases the effectiveness about 8%.

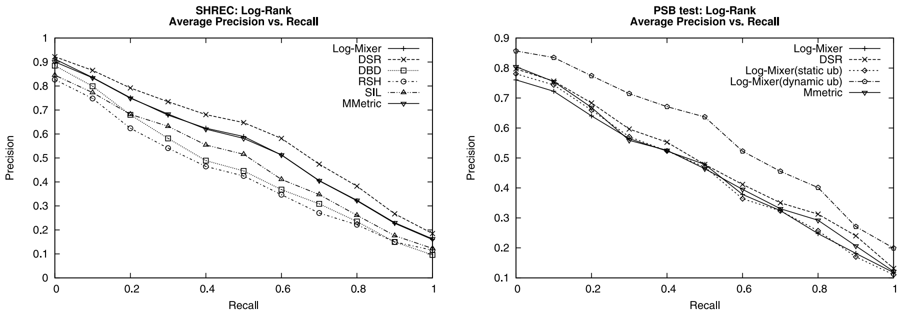

Fig. 3. Effectiveness of log-rank. Left: Shrec Dataset, right: PSB test.

Table 3. Effectiveness of Fixed Rank-Mixer. Left: combination of FVs, right: combination of the results of Chaouch and Lian.

Method	**NN**	**1T**	**2T**	**E**
log-rank	0.863	0.504	0.640	0.442
Fixed log-rank ($k = 32$)	0.850	0.496	N.A.	0.428
Fixed log-rank ($k = 34$)	0.850	0.495	N.A.	0.425
Fixed log-rank ($k = 36$)	0.850	0.490	0.611	0.427
Fixed log-rank ($k = 38$)	0.850	0.489	0.613	0.427
Fixed log-rank ($k = 40$)	0.863	0.490	0.617	0.428

Method	**NN**	**1T**	**2T**	**E**
M(Chaouch, Lian)	0.950	0.788	0.906	0.642
Fixed log-rank ($k = 32$)	0.950	0.774	N.A.	0.640
Fixed log-rank ($k = 34$)	0.950	0.774	N.A.	0.639
Fixed log-rank ($k = 36$)	0.950	0.775	0.901	0.641
Fixed log-rank ($k = 38$)	0.950	0.776	0.899	0.639
Fixed log-rank ($k = 40$)	0.950	0.776	0.899	0.638

The above results where computed using the whole rank, that is $k = |\mathbb{U}|$. However, we can compute some statistics for fixed k, such as the Nearest Neighbor or the E-Measure given that $k \geq 32$. If we test it on the SHREC dataset, we could also compute the R-Precision if $k \geq 18$ and the Bull-Eye Percentage if $k \geq 36$. For the PSB, we would have to take $k \geq 50$ just to compute the First Tier. Relying on the same basis of the definition of the E-Measure that a user is interested just in the first screen of results, we will take $k \geq 32$ for our tests, and we will test it on the SHREC dataset. Table 3 shows the results, where "N.A." means not applicable. The function we used in these tests was $f(x) = \log(200) - \log(x)$.

These results show that there is no need of having the complete rank, it is enough to have approximately the 40 first elements to get an improvement close to the one obtained using the whole rank.

Rank-Booster. In our tests, the *selectElements* function of the Rank-Booster algorithm selects the first $k(k-1)/2$ elements. Motivated by the results of the previous experiments, we took $f^+(x, y) = \log(2k_b + 1) - \log(x + y + 1)$.

Before performing the tests, we had to compute the best value for k_b. For doing so we computed the R-Precision of the Rank-Booster with different values of k_b. We only used the DSR feature vector because we wanted to choose a value of k_b independent of the used methods. It is important to notice that $k_b = 1$ is the same as not using any improvement method over the descriptor.

Figure 4 shows that $k_b = 13$ is the best choice for SHREC and that every $3 \leq k_b \leq 16$ yields to improvement in the effectiveness of the method. The figure

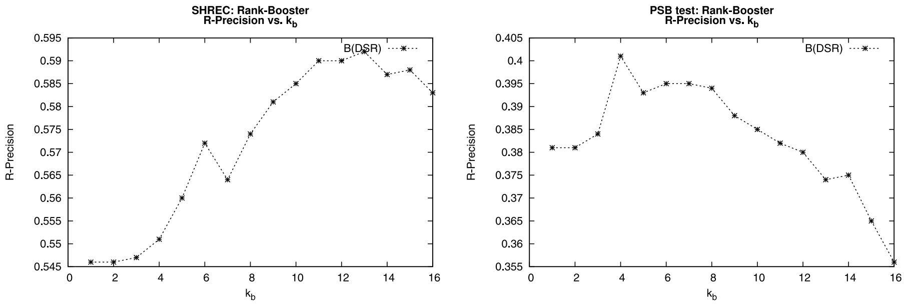

Fig. 4. R-Precision of Rank-Booster. Left: Shrec Dataset, right: PSB test.

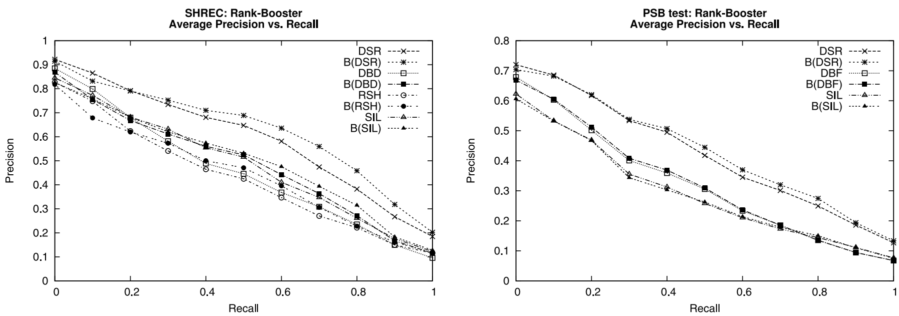

Fig. 5. Effectiveness of Rank-Booster. Left: Shrec Dataset, right: PSB test.

also shows that the best possible value for PSB is $k_b = 6$. It also follows that any $0 < k_b \leq 9$ yields to improvements in the effectiveness or maintains it.

Figure 5, show the precision-recall curves of the retrieval methods used in SHREC and the best methods in PSB. Table 2(right) show the details of the results for the SHREC dataset. These results show that the Rank-Booster method applied over the DSR descriptor gives better results that the ones obtained in SHREC 09 [1] using the global-local approach. Rank-Booster increases effectiveness up to 8%. This is an extremely good result, because this method could be applied to any existing framework, without the need of having several retrieval methods. Although we have devised no way to efficiently compute the optimal value for k_b, it follows from the results that, it suffices to take a small k_b to achieve higher effectiveness.

Combination of Rank-Mixer and Rank-Booster. Table 4 shows that combining Rank-Mixer and Rank-Booster leads to a further improvement of the effectiveness. Combining the methods increases the effectiveness about 2% with respect to the Rank-Mixer.

Table 4. Effectiveness of combining Rank-Mixer and Rank-Booster

Method	NN	1T	2T	E	DCG
Mixer	0.863	0.504	0.640	0.442	0.738
Mixer(B)	0.812	0.511	0.649	0.451	0.786
Booster(Mixer)	0.863	0.517	0.656	0.455	0.782
Booster(Mixer(B))	0.812	0.497	0.633	0.440	0.771

Table 5. Results of ImageCLEF@ICPR Fusion Task. Left: original methods, right: fusion of methods.

Method	MAP	P5	P10
Text1	0.35	0.58	0.56
Text2	0.35	0.65	0.62
Text3	0.43	0.7	0.66
Text4	0.38	0.65	0.62
Visual1	0.01	0.09	0.08
Visual2	0.01	0.08	0.07
Visual3	0.01	0.09	0.07
Visual4	0.01	0.09	0.08

Method	MAP	P5	P10
T(2,3,4) V3 (8.41)	0.491	0.760	0.696
T(2,3,4)	0.480	0.704	0.672
T(1,2,3,4)	0.474	0.712	0.648
T(1,2,3)	0.473	0.712	0.664
T(1,2,3,4) V(234)	0.466	0.752	0.676
T(1,2,3,4) V(134)	0.464	0.744	0.692
T(1,2,4)	0.464	0.688	0.640
T(1,2,3,4) V(1,2,3)	0.451	0.744	0.688

ImageCLEF@ICPR Fusion Task Results. In this task [10] we had to fusion textual and visual results. As not all methods returned the same number of elements we slightly modified our algorithm. We use the function $f^{+}(x) = T - \log_2(x)$, with $T = 8.0$ fixed for all except the first result. The first result is the best static combination of the methods with the best possible value of $T = 8.41$. The best result yields an improvement of 14% and the best fully automatic combination yields an improvement of 11%.

5 Conclusions

We presented two algorithms for increasing the effectiveness of multimedia retrieval methods. One of these algorithms, the log-rank, outperforms the state of the art rank fusion methods MedRank and Reciprocal Rank. The other algorithm can not be compared against these state of the art methods because it only uses one single method to improve its effectiveness. One important advantage of the proposed methods is that they do not rely on metric retrieval methods, and they can be applied over any method that generates a ranking of the elements given a query object. Another advantage of these methods is that they can be directly applied on top of the indexing scheme of the used methods, without the need of building a custom indexing scheme. An additional advantage of the proposed methods is that one does not need to normalize the databases nor the multimedia descriptors, as required by the multimetric approach.

In the future work, we will study the problem of estimating the parameter k_b of the Rank-Booster method. We will also research how to select the retrieval methods to use in order to get the effectiveness of Rank-Booster and Rank-Mixer closer to the upper bound showed in Section 4.1.

Acknowledgments

We want to thank to Afzal Godil for kindly giving us the best ranks of the SHREC contest. Finally, we would like to thank to Mohamed Chaouch, Zhouhui Lian, Thibault Napoléon and their respective teams, for letting us to use their results in our investigation.

References

1. Akgül, C., et al.: Shrec 2009 track: Generic shape retrieval. In: Proc. Eurographics 2009 Workshop on 3D Object Retrieval (3DOR), pp. 61–68. Eurographics Association (2009)
2. Böhm, C., Berchtold, S., Keim, D.A.: Searching in high-dimensional spaces: Index structures for improving the performance of multimedia databases. ACM Computing Surveys 33(3), 322–373 (2001)
3. Bustos, B., Keim, D., Saupe, D., Schreck, T., Vranić, D.: Using entropy impurity for improved 3D object similarity search. In: Proc. IEEE International Conference on Multimedia and Expo (ICME), pp. 1303–1306. IEEE, Los Alamitos (2004)
4. Bustos, B., Skopal, T.: Dynamic similarity search in multi-metric spaces. In: Proc. ACM SIGMM International Workshop on Multimedia Information Retrieval (MIR), pp. 137–146. ACM Press, New York (2006)
5. Chávez, E., Navarro, G., Baeza-Yates, R., Marroquin, J.: Searching in metric spaces. ACM Computing Surveys 33(3), 273–321 (2001)
6. Cormack, G.V., Clarke, C.L.A., Büttcher, S.: Reciprocal rank fusion outperforms condorcet and individual rank learning methods. In: Proc. Annual International ACM Conference on Research and Development in Information Retrieval (SIGIR), poster (2009)
7. Fagin, R., Kumar, R., Sivakumar, D.: Efficient similarity search and classification via rank aggregation. In: Proc. ACM SIGMOD International Conference on Management of Data (SIGMOD), pp. 301–312. ACM, New York (2003)
8. Fang, R., Godil, A., Li, X., Wagan, A.: A new shape benchmark for 3D object retrieval. In: Bebis, G., Boyle, R., Parvin, B., Koracin, D., Remagnino, P., Porikli, F., Peters, J., Klosowski, J., Arns, L., Chun, Y.K., Rhyne, T.-M., Monroe, L. (eds.) ISVC 2008, Part I. LNCS, vol. 5358, pp. 381–392. Springer, Heidelberg (2008)
9. Ho, T.K., Hull, J.J., Srihari, S.N.: On multiple classifier systems for pattern recognition. In: Proc. International Conference on Pattern Recognition (ICPR), pp. 66–75 (1992)
10. Müller, H., Kalpathy-Cramer, J.: The ImageCLEF medical retrieval task at ICPR 2010 - information fusion to combine visual and textual information. In: Ünay, D., Çataltepe, Z., Aksoy, S. (eds.) ICPR 2010. LNCS, vol. 6388, pp. 101–110. Springer, Heidelberg (2010)
11. Shilane, P., Min, P., Kazhdan, M., Funkhouser, T.: The princeton shape benchmark. In: Shape Modeling International (2004)
12. Suen, C.Y., Lam, L.: Multiple classifier combination methodologies for different output levels. In: Kittler, J., Roli, F. (eds.) MCS 2000. LNCS, vol. 1857, pp. 52–66. Springer, Heidelberg (2000)
13. Vranic, D.V.: Desire: a composite 3D-shape descriptor. In: IEEE International Conference on Multimedia and Expo. (ICME), pp. 962–965 (2005)
14. Zezula, P., Amato, G., Dohnal, V., Batko, M.: Similarity Search: The Metric Space Approach (Advances in Database Systems). Springer, Heidelberg (2005)