Learning to Weight Color and Depth for RGB-D Visual Search

Alioscia Petrelli(✉) and Luigi Di Stefano

University of Bologna, Bologna, Italy
{alioscia.petrelli,luigi.distefano}@unibo.it
http://vision.deis.unibo.it

Abstract. Both color and depth information may be deployed to seek by content through RGB-D imagery. Previous works dealing with global descriptors for RGB-D images advocate a decision level fusion whereby independently computed color and depth representations are juxtaposed to pursue similarity search. Differently, in this paper we propose a *learning-to-rank* paradigm aimed at weighting the two information channels according to the specific traits of the task and data at hand, thereby effortlessly addressing the potential diversity across applications. In particular, we propose a novel method, referred to as *kNN-rank*, which can learn the regularities among the outputs yielded by similarity-based queries. A further novel contribution of this paper concerns the *Hyper-RGBD* framework, a set of tools conceived to enable seamless aggregation of existing RGB-D datasets in order to obtain new data featuring desired peculiarities and cardinality.

Keywords: RGB-D image search · Compact descriptors · Learning-to-rank

1 Introduction

Encoding image content into compact though distinctive representations is key to retrieval performance in large-scale visual search. To pursue visual search one would typically match the query image against those stored in a database by comparing global image representations, so as to receive the digital content linked to the most similar one. In this realm, numerous works, such as [4,14,18], address how to represent images by short binary codes conducive to efficient matching and storage when dealing with large-size databases.

Reliance on compact binary representations is an essential trait in mobile visual search alike. Here, the image acquired by a mobile device's camera is encoded and transmitted via a wireless network to a remote server undertaking database search. Therefore, bandwidth constraints mandate the images sent to the server to be represented as compactly as possible. How to design an effective mobile visual search architecture leveraging on compact image representations has been addressed in several research papers [4,7,9] as well as in the recently defined *Compact descriptors for visual search* (CDVS) standard by the MPEG group.

S. Battiato et al. (Eds.): ICIAP 2017, Part I, LNCS 10484, pp. 648–659, 2017.
https://doi.org/10.1007/978-3-319-68560-1_58

Similar technology trends and research challenges are likely to become increasingly relevant in the field of RGB-D imagery. Indeed, broad diffusion of consumer depth cameras has enabled the creation of a few relatively large-size RGB-D datasets comprising thousands or tens of thousands images. Moreover, mobile devices start being endowed with the ability to sense depths, either by mountable cameras, like *Structure* by *Occipital*, or fully integrated sensors, e.g. as provided by Google's *Project Tango* technology which, in particular, is on the verge of deployment in off-the-shelf smartphones. Hence, one might be lead to foresee more and more large RGD-D datasets to become available as well as the emergence of applications performing Visual Search via RGB-D images taken by mobile devices. The above trends, thus, are likely to foster considerable research efforts towards the novel topic of compact binary representations for RGB-D visual search.

The work described in [15,16] proposes the first investigation on how to globally represent RGB-D images by compact binary codes. The experimental analyses reveal that encoding of depths is key to recognize object categories, whereas object instances are mainly told apart based on RGB information. More interestingly, though, the authors highlight how different tasks and datasets exhibit different peculiarities, so that, in general, naively chaining together the binary codes associated with color and depth yields sub-optimal performance. Rather, an effective approach to RGB-D visual search should pursue automatic learning of the relative prominence of color and depth in the addressed scenario.

In information retrieval, the *learning-to-rank* paradigm provides a sound framework to combine different strategies by learning a model that fuses into a joint ranking the individual rankings yielded independently by the different strategies. Learning-to-rank approaches perform a supervised learning aimed at discovering which strategies produce better rankings in the addressed scenario and, accordingly, learn how to weight properly the individual rankings into the final one. Such paradigm has been deployed in Content-Based Image Retrieval[1] to weight the contributions of different feature kinds extracted from RGB images.

In this paper, we propose the first investigation dealing with application of the learning-to-rank paradigm to RGB-D visual search by binary codes. In particular, we propose and apply to the architecture described in [16] a novel learning-to-rank approach, dubbed *kNN-rank*. This approach tries to obtain a joint ranking for the given query by learning the regularities within the k-NNs retrieved by matching color and depth codes, such regularities concerning both the types of object found as neighbors as well as the associated distances. Intuitively, if we query by a yellow cup we might retrieve cups based on depth and bananas based on color, so that we would wish to learn to ignore the color channel when aiming at category recognition while positively weighting it when willing to recognize that specific cup.

Although a few relatively large RGB-D datasets are available nowadays, their size is far smaller than that of state-of-the-art RGB datasets. To facilitate

[1] Here, unlike visual search, the task is to provide the user several images similar to the query.

experimentation with larger and diverse datasets, a second novel contribution of this paper concerns a software framework, referred to as *HyperRGBD*, that allows researchers to create straightforwardly new data with desired traits and peculiarities by mixing arbitrarily and seamlessly images drawn from different RGB-D datasets.

2 Previous Work

In the last few years, many papers have addressed the task of object recognition from RGB-D images. Most of them [2,8,17] fuse color and depth data at feature level through either hand-crafted descriptors or deep learning approaches. Such rich representations are then fed to a classifier (e.g. a SVM) trained to recognize the content of the query image. Differently, in [3] depth and color information are fused at decision level. Indeed, both color and depth are represented by eight different descriptors and a specific SVM is trained for each feature type and object category. The final decision is taken by a neural network presented with the output of all these SVMs.

Other works are focused on how to weight the contribution of color and depth as well as of diverse shape cues. [11] adopts an AdaBoost learning procedure to weight color and depth for the task of face recognition, whereas [13] analyzes different strategies for weighting five different 3D descriptors on the *Princeton Shape Benchmark*. In [1], Bar-Hillel et al. propose the O^2NBNN framework that describes images through multiple channels encoding intensity, depth information or a feature level fusion of the two contributions. At training time, an optimization allows for learning the proper weights for each class and channel that are, then, used to predict the object class from the query image. However, all the above mentioned methods rely on rich, high-dimensional descriptors and leverage on classifiers, while in the realm of visual search one would typically rely on compact representations and perform a similarity search across the database.

Learning-to-rank has been effectively applied in RGB-based image retrieval. [12] quantitatively compares three different approaches (pointwise, pairwise and listwise) on four datasets. The work in [6] applies and compare *Ranking SVM*, Genetic algorithms and Association Rules for ranking eighteen types of descriptors (color, texture and shape based) on two RGB datasets. To the best of our knowledge, the only work that exploits a learning-to-rank paradigm to fuse color and depth data has been recently described in [5]. The method measures the similarity between a query and a reference image by means of an ensemble of dense matchings that weight differently the features extracted from color and depth data. Then, the scores obtained by dense matchings are ranked through *Ranking SVM* [10]. However, this approach is not conceived for large-scale visual search but to re-rank a set of candidates priorly identified by a classifier, such as the algorithm proposed in [17]. Moreover, it would not be applicable to mobile scenarios due to the requirement of sending to the remote server the full RGB-D image rather than just a compact binary code.

3 Visual Search Architecture

In this section we outline the visual search architecture proposed in [16] and deployed in this paper to apply learning-to-rank methods for RGB-D image search. First, a set of patches are extracted densely from the query RGB-D image and described through Kernel Descriptors. In particular, the appearance information associated with each patch is represented by kernels dealing with intensity gradients (KD_I) and color (KD_C), while 3D shape information is captured by kernels encoding depth gradients (KD_D) and *Spin Images* descriptors (KD_S). Then, these local features are aggregated into a global image description by *Fisher Kernel*. Finally, the *Spherical hashing* algorithm provides the compact binary code used to carry out similarity search within the image database. The experimental analysis in [16] highlights that the information extracted from the depth and color images should better be aggregated at decision rather than feature level. Accordingly, the four Kernel Descriptors (KD_I, KD_C, KD_D, KD_S) are computed, aggregated and hashed separately, so as to end up with four binary codes, referred to as B_I, B_C, B_D and B_S, that are simply juxtaposed to create the final tag, B, deployed to seek for the most similar image within the database. Then, an object instance (category) gets recognized correctly if the most similar database image retrieved by matching the binary tag comes from the same instance (category) as the query image. Comparison between binary tags is achieved by the fast Hamming distance and the search performed efficiently by indexing the database through the *multi-probe LSH* scheme. Finally, the matching process is robustified by the weighted k-NN classifier ($k = 9$).

However, as highlighted in [15,16], simple juxtaposition of the binary codes hardly succeeds in capturing the diverse distinctiveness that the deployed feature channels may convey in different tasks and datasets. Accordingly, the next section describes an approach aimed at learning to weight the relative contributions of the individual binary codes in order to seamlessly adapt the pipeline to the peculiarities of the addressed scenario.

4 The kNN-rank Approach

Figure 1 allows for visualizing the results of a query carried out on the *Washington* dataset by the visual search architecture proposed in [16]. It can be observed that the binary codes dealing with depth information (B_D, B_S) succeed in identifying the correct category, whilst this is not the case of those extracted from the RGB image (B_I, B_C). In particular, matching based on color (B_C) mistakes the *bowl* for a *cup* due to the very similar texture patterns. This, in turn, hinders the final matching based on the juxtaposed codes, B, which returns a wrong category (i.e. *cup*).

However, had we be presented with these results and been told to trust depth much more than color, we would have been able to pick the correct category. Similar observations drawn from analyzing the results of several queries lead us to the intuition that the information conveyed by retrieved images contains

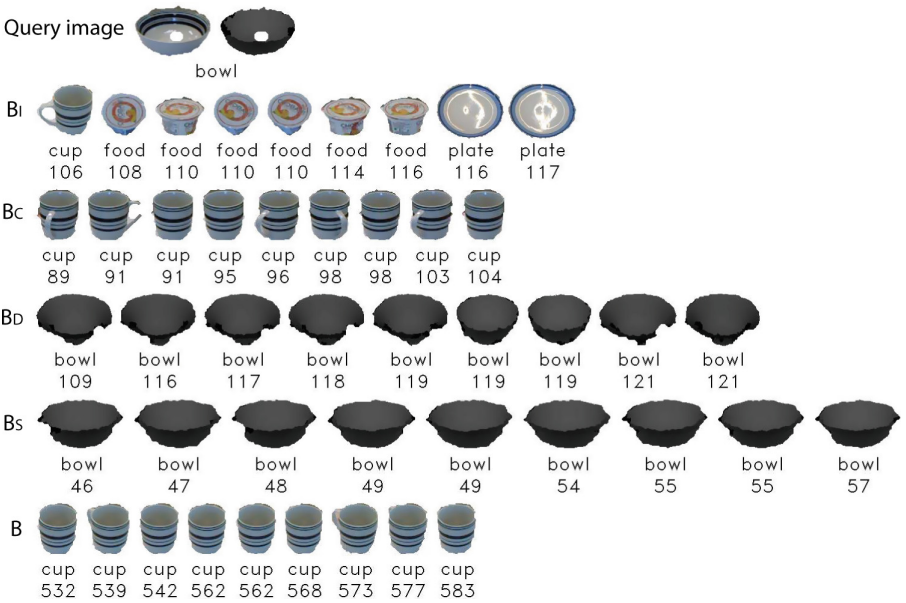

Fig. 1. Result of a query on the *Washington* dataset. The first row reports the query RGB-D image together with its associated category ("bowl"). The next four rows show the $k = 9$ most similar images according to the four binary codes (B_I, B_C, B_D and B_S). The last row shows the $k = 9$ images retrieved by the binary tag, B. The category and Hamming distance from the query image are shown below each retrieved image.

regularities that may be exploited in order to learn how to make decisions aware of the specific scenario and data.

Accordingly, this section describes a novel learning-to-rank method, dubbed *kNN-rank*, that, given the results of a query, defines a set of feature vectors based on both the labels (either instance or category labels, depending on the recognition task) and distances of retrieved images. In particular, a feature vector is created for each different retrieved label, such feature vectors used at training time to learn a ranking function while at test time to rank the label with respect to the query.

More in detail, the labels relevant to a query are those retrieved either by each of the individual binary codes or by juxtaposing them. For example, for the query illustrated in Fig. 1, the relevant labels are "cup", "food", "plate" and "bowl". Then, given a relevant label, l_i, an associated feature vector, x_i, is assembled by computing a pair of features for each retrieved image. The first feature in the pair encodes whether the corresponding image is labeled as l_i or not: in the former case, it is "fired" and equal to the measured Hamming distance, in the latter it is set to zero. Considering the exemplar query of Fig. 1 and label "cup", the first feature of the pair for each retrieved image is shown in blue on the left side of Fig. 2. Conversely, the second feature is fired, i.e. equal

106	0	0	0	0	0	0	0	0	0	108	110	110	110	114	116	116	117
89	91	91	95	96	98	98	103	104	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	109	116	117	118	119	119	119	121	121
0	0	0	0	0	0	0	0	0	46	47	48	49	49	54	55	55	57
532	539	542	562	562	568	573	577	583	0	0	0	0	0	0	0	0	0

Fig. 2. Feature vector produced by the *kNN-rank* method for label "cup" according to the query results depicted in Fig. 1. Each row deals with the images retrieved based on a different code (i.e. B_I, B_C, B_D, B_S, B) and consists of $2 \times k$ elements. "Blue" features encode the Hamming distance for "cup" images whereas "green" features the Hamming distance for "non-cup" images. (Color figure online)

to the measured Hamming distance, for all the retrieved images showing labels other than l_i. Considering again the query of Fig. 1 and label "cup", the second feature of the pair for each retrieved image is shown in green on the right side of Fig. 2.

Similarly to the Ranking SVM approach [10], we solve a binary classification problem. More precisely, at training time we randomly select N images from the database to be treated as queries. For each query, we apply a k-nn search in the database based on B_I, B_C, B_D, B_S and B. As described, we create a feature vector for each relevant label, l_i, and then assign either $+1$ or -1 to each feature vector based on whether l_i is correct or wrong for the query. These samples are normalized to similarity scores in the interval $[0, 1]$ and used to train a linear SVM. In particular, denoted as $x_i\,(d)\,,\ d \in \{I, C, D, S\}$, the Hamming distances associated with the four binary codes, the corresponding normalized features are given by

$$\tilde{x}_i(d) = \frac{\max\limits_i x_i(d) - x_i(d)}{\max\limits_i x_i(d)} \tag{1}$$

At test time, given a query, each relevant label l_i is ranked with respect to the query according to the score computed by the trained SVM:

$$f(l_i) =< w, x_i > \tag{2}$$

5 The HyperRGBD Framework

This section outlines a C++ software framework, referred to as *HyperRGBD*, devised to enable researchers and practitioners to build effortlessly new datasets by aggregating images from different existing RGB-D datasets. For example, one might wish to experiment with datasets larger than existing ones, which would seamlessly be attainable by deploying *HyperRGBD* to aggregate the images belonging to existing datasets into a larger data corpus. Furthermore, should a dataset be biased towards certain abundant categories with others featuring a few samples only, it would be just as seamless to build a more balanced

dataset by using *HyperRGBD* to draw samples for the rare categories from other datasets. Another example may deal with changing the granularity of categories, e.g. aggregating "chair", "table" and "couch" into a broader "furniture" category or splitting "fruit" into more specific categories like "apple", "orange" and "banana". At present, we have integrated in the framework the main existing RGB-D datasets for object recognition, i.e. *Washington*, *CIN 2D+3D*, *BigBIRD* and *MV-RED*, that are briefly described in a project page[2] we make available, along with the source code of the framework, so to foster research activity on perception from RGB-D imagery and enable researchers to integrate their data.

We exploited *HyperRGBD* to obtain two new RGB-D datasets used in our experiments besides the main existing ones. We aggregated the above four datasets to create two new datasets and tested both in instance as well as category recognition scenarios. The former, *HyperRGBD*, merges all the available images. The latter, *HyperRGBD - Balanced*, addresses the wide differences in size between existing datasets by balancing them upon aggregation. More precisely, for instance recognition scenario, we identify the dataset with the fewest instances (*BigBIRD* comprising 114 instances) and level down the others by randomly selecting 114 instances per dataset. In the case of category recognition, instead, for each of the categories of the aggregated dataset, we search for the dataset providing the smallest amount of instances and, accordingly, populate the category by randomly selecting that amount of instances from each dataset. Once the datasets are gathered, both for category and instance recognition, a tenth of the dataset is used as test set and the remaining to perform the training. The procedure is repeated 10 times on different randomly generated test sets so to obtain 10 different trials.

6 Experimental Evaluation

To assess the ability of the novel *kNN-rank* method to properly weight color and depth channels across different tasks and data, our experimental evaluation compares it against the *SVMrank* approach proposed in [10], a *Ranking SVM* formulation that has proved to be effective in a variety of real settings. As a baseline, we also include in the evaluation the matching of juxtaposed binary codes encoding depth and color information, as delineated in previous work dealing with RGB-D visual search [15,16].

In the experiments reported in this section, queries and database images are encoded by allocating 512 bits to each of the four binary codes (B_I, B_C, B_D, B_S), so that the final tag, B, gets as large as 2048 bits. Indeed, extensive experimental investigation showed that longer descriptions would not provide significant improvement in the recognition capability of the architecture. Furthermore, even though recognition rates decrease as the description length decreases, the ranking between the approaches considered in this section remains identical. We also report the recognition performance achieved by individually matching binary

[2] http://www.vision.disi.unibo.it/research/78-cvlab/107-hyperrgbd.

codes B_I, B_C, B_D, B_S, which in these kinds of experiment are given the same length (2048 bits) as B.

In the *SVMrank* approach settings, the feature vectors x_i are four-dimensional and consist of the four Hamming distances between the binary codes B_I, B_C, B_D, B_S computed from query and database images as described in Sect. 3. To perform the training we randomly select N images from the database to be treated as queries; then, for each query, we randomly pick 500 relevant images (i.e. for which the category/instance is the same of the query image) and equally many irrelevant ones, so as to create pairs of feature vectors x_i, x_j dealing with the same query in which one is associated to a relevant image and the other to an irrelevant one. Thereby, the binary classifier can be provided with training samples according to the standard formulation of the *Ranking SVM* approach. At query time we avoid the computation of the ranking score for all database images and instead rank only a subset of candidates. Purposely, we individually match the four binary codes B_I, B_C, B_D, B_S to identify, for each, the k most similar images. Moreover, we match the tag given by juxtaposing the four binary codes, B, to retrieve equally many images. The final set of candidates is the union of these retrieved images (i.e., at most $k \times 5$ images).

Both for *SVMrank* and *kNN-rank*, to perform training, we extract $N = 2000$ images treated as queries and, as suggested in [16], similarity searches have been performed by setting $k = 9$ for all the methods.

6.1 Results

Table 1 summarizes all the results obtained by our quantitative evaluation on all the available datasets in both category and instance recognition tasks. We evaluate performance based on the recognition rate, i.e. top-1 accuracy, as this is the standard metric concerning visual search scenarios, where one would wish

Table 1. Recognition rates obtained on the considered datasets by matching the binary codes B_I, B_C, B_D, B_S, B, by a learning-to-rank approach (*SVMrank*) and by our novel proposal (*kNN-rank*).

	Dataset	B_D	B_S	B_I	B_C	B	SVMrank	kNN-rank
Category recognition	Washington	0.568	0.550	0.531	0.450	0.791	0.778	0.801
	CIN 2D+3D	0.666	0.597	0.616	0.562	0.768	0.764	0.771
	MV-RED	0.494	0.586	0.550	0.578	0.760	0.743	0.769
	HyperRGBD - Balanced	0.518	0.521	0.500	0.462	0.732	0.719	0.748
	HyperRGBD	0.539	0.557	0.525	0.505	0.773	0.751	0.785
Instance recognition	Washington	0.342	0.393	0.489	0.845	0.843	0.882	0.879
	CIN 2D+3D	0.606	0.503	0.723	0.795	0.834	0.840	0.841
	MV-RED	0.434	0.597	0.686	0.957	0.929	0.960	0.960
	BigBIRD	0.281	0.361	0.434	0.822	0.727	0.814	0.820
	HyperRGBD - Balanced	0.434	0.493	0.619	0.896	0.831	0.884	0.894
	HyperRGBD	0.413	0.482	0.612	0.916	0.863	0.908	0.912
	Average:	0.481	0.513	0.571	0.708	0.805	0.822	0.834

to receive information linked to image content[3]. This is the metric adopted in [15,16] as well as in most previous work related to instance/category recognition from RGB-D imagery [2,5,8,17]. Each row reports the recognition rates obtained by the considered approaches on a different dataset and type of experiment (i.e. either category or instance recognition). The adopted color code allows for perceiving clearly the differences in performance as higher recognition rates are denoted by darker background colors within cells.

The comparison between the results obtained by separate deployment of the different cues (B_I, B_C, B_D, B_S) and concatenation of descriptors, B, confirms the findings already discussed in Sect. 3. As a matter of fact, fusing descriptions is clearly beneficial for category recognition, whereas, in general, much less effective to tell apart specific object instances. In the latter task, indeed, performance depends quite significantly on the specific type of data, with juxtaposition providing higher recognition rate in the *CIN 2D+3D* dataset and turning out useless with the type of objects included in the *Washington* dataset, where description based on color (B_C) suffices in delivering the highest performance. On the remaining datasets, juxtaposing representations (B) is even detrimental with respect to allocating all the available bits to color (B_C). Thus, although the simple recognition strategy based on matching juxtaposed descriptors delineated in [15,16] is overall effective, as vouched by the average figures across the first five columns (B_I, B_C, B_D, B_S, B) reported in the last row of Table 1, it turns out clearly sub-optimal in many relevant settings.

The *SVMrank* approach partly addresses such issue by providing, generally, higher recognition rates, as reported by the average recognition rate in the last row. Nonetheless, even if the method properly deals with instance recognition tasks by providing top recognition rates on all the datasets, a comparison limited to the category recognition task between *SVMrank* and B shows slightly better results in favor of the latter. Such behaviour could be ascribed to the large intra-class variability of the objects belonging to a category which renders the task more challenging than telling apart a specific object from others. *SVMrank* may not be powerful enough to learn the regularities that tie the objects of a same category.

That is not the case of the novel *kNN-rank* method introduced in this paper, that, as vouched by the last column of Table 1, can yield recognition rates higher than B also in category recognition experiments, behaves effectively on both the tasks and correctly adapts to all the datasets. The background color code permits to catch at a glance that our proposal provides the highest recognition rates on most of the datasets and ties on the others. Again, the average figures on the last row show the overall superiority of *kNN-rank*. Hence, we can conclude that learning the regularities underlying retrieved images is an effective strategy for obtaining correct rankings.

Table 1 reports also the results on the two new datasets created through the *HyperRGBD* framework. The results are coherent with those obtained on the

[3] Differently, in image retrieval, one is interested in receiving several images and therefore top-n accuracy is adopted.

individual datasets and highlight the good scalability of learning-to-rank methods to larger datasets. It is worth showing that *HyperRGBD* and *HyperRGBD - Balanced* are genuinely new datasets and not plain aggregations of the constituent datasets. As evidence of that, Fig. 3 reports three examples of queries performed on the *HyperRGBD - Balanced* dataset by matching the binary tag, B. The retrieved images belong to different datasets. Furthermore, in the first two examples, even though the query images belong to the *Washington* dataset, two images from the *CIN 2D+3D* are returned as top-1 result. These examples show that the *HyperRGBD* framework mixes datasets effectively and prove that the recognition rates reported in Table 1 for the *HyperRGBD* and *HyperRGBD - Balanced* are not the mere averages of the results already obtained on the other datasets.

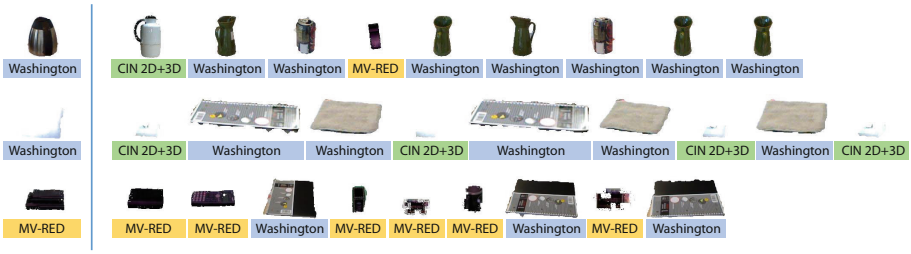

Fig. 3. Result of three queries on the *HyperRGBD - Balanced* dataset. On the left we show the query images, whereas on the right the $k = 9$ images retrieved by matching the binary tag, B. Each image is labeled with the dataset it comes from.

7 Final Remarks

This paper shows that applying the learning-to-rank paradigm for weighting color and depth cues in RGB-D visual search does improve performance significantly and, in particular, allows for handling seamlessly diverse datasets and tasks. This is achieved by applying the novel *kNN-rank* method, that analyses the regularities in the retrieved images so as to learn the contribution conveyed by the different cues. The approach provides top performance on all the experiments we performed, both on the main existing RGB-D datasets as well as on two new datasets we created by means of the proposed *HyperRGBD* framework.

Although the *kNN-rank* method has been applied to Hamming distances of binary codes encoding color and depth cues, nothing indicates that the approach could not be successfully deployed in other contexts. So far, learning to rank methods have been applied in Content-Based Image Retrieval wherein large-scale RGB databases are encoded by numerous color, shape and texture features. Thus, we plan to test and evaluate our proposal in these settings so as to assess the ability of *kNN-rank* to scale to databases comprising million of images and to properly weight a larger number of cues.

References

1. Bar-Hillel, A., Hanukaev, D., Levi, D.: Fusing visual and range imaging for object class recognition. In: International Conference on Computer Vision, pp. 65–72 (2011)
2. Blum, M., Wulfing, J., Riedmiller, M.: A learned feature descriptor for object recognition in RGB-D data. In: International Conference on Robotics and Automation, pp. 1298–1303 (2012)
3. Browatzki, B., Fischer, J.: Going into depth: evaluating 2D and 3D cues for object classification on a new, large-scale object dataset. In: International Conference on Computer Vision Workshops (2011)
4. Chandrasekhar, V., Lin, J., Morere, O., Veillard, A., Goh, H.: Compact global descriptors for visual search. In: Data Compression Conference, pp. 333–342 (2015)
5. Cheng, Y., Cai, R., Zhang, C., Li, Z., Zhao, X., Huang, K., Rui, Y.: Query adaptive similarity measure for RGB-D object recognition. In: International Conference on Computer Vision, pp. 145–153 (2015)
6. Faria, F.F., Veloso, A., Almeida, H.M., Valle, E., Torres, R.D.S., Gonçalves, M.A., Meira, W.: Learning to rank for content-based image retrieval. In: International Conference on Multimedia Information Retrieval (2010)
7. Guan, T.A.O., Wang, Y., Duan, L., Ji, R.: On-device mobile landmark recognition using binarized descriptor with multifeature fusion. Trans. Intell. Syst. Technol. **7**(1), 12–29 (2015)
8. Gupta, S., Girshick, R., Arbeláez, P., Malik, J.: Learning rich features from RGB-D images for object detection and segmentation. In: Fleet, D., Pajdla, T., Schiele, B., Tuytelaars, T. (eds.) ECCV 2014. LNCS, vol. 8695, pp. 345–360. Springer, Cham (2014). doi:10.1007/978-3-319-10584-0_23
9. He, J., Feng, J., Liu, X., Cheng, T., Lin, T.H., Chung, H., Chang, S.F.: Mobile product search with bag of hash bits and boundary reranking. In: Conference on Computer Vision and Pattern Recognition, pp. 3005–3012 (2012)
10. Joachims, T.: Training linear SVMs in linear time. In: International Conference on Knowledge Discovery and Data Mining (2006)
11. Li, S.Z., Zhao, C.S., Ao, M., Lei, Z.: Learning to fuse 3D+2D based face recognition at both feature and decision levels. In: Zhao, W., Gong, S., Tang, X. (eds.) AMFG 2005. LNCS, vol. 3723, pp. 44–54. Springer, Heidelberg (2005). doi:10.1007/11564386_5
12. Li, Y., Zhou, C., Geng, B., Xu, C., Liu, H.: A comprehensive study on learning to rank for content-based image retrieval. Sig. Process. **93**(6), 1426–1434 (2013)
13. Lv, T., Liu, G., Huang, S.B., Wang, Z.X.: Selective feature combination and automatic shape categorization of 3D models. In: International Conference on Fuzzy Systems and Knowledge Discovery, pp. 447–451 (2009)
14. Perronnin, F., Liu, Y., Jorge, S.: Large-scale image retrieval with compressed fisher vectors. In: Conference on Computer Vision and Pattern Recognition, pp. 3384–3391 (2010)
15. Petrelli, A., Pau, D., Stefano, L.: Analysis of compact features for RGB-D visual search. In: Murino, V., Puppo, E. (eds.) ICIAP 2015. LNCS, vol. 9280, pp. 14–24. Springer, Cham (2015). doi:10.1007/978-3-319-23234-8_2

16. Petrelli, A., Pau, D., Plebani, E., Di Stefano, L.: RGB-D visual search with compact binary codes. In: International Conference on 3D Vision, pp. 82–90 (2015)
17. Socher, R., Huval, B., Bhat, B., Manning, C.D., Ng, A.Y.: Convolutional-recursive deep learning for 3D object classification. In: Advances in Neural Information Processing Systems, pp. 1–9 (2012)
18. Song, D., Liu, W., Ji, R., Meyer, D.A., Smith, J.R.: Top rank supervised binary coding for visual search. In: International Conference on Computer Vision, pp. 1922–1930 (2015)