Scale-free content based image retrieval (or nearly so)

Adrian Popescu, Alexandru Ginsca, Hervé Le Borgne
CEA, LIST, Laboratory of Vision and Content Engineering,
F-91191 Gif-sur-Yvette, France
{adrian.popescu,alexandru.ginsca,herve.le-borgne}@cea.fr

Abstract

When textual annotations of Web and social media images are poor or missing, content-based image retrieval is an interesting way to access them. Finding an optimal trade-off between accuracy and scalability for CBIR is challenging in practice. We propose a retrieval method whose complexity is nearly independent of the collection scale and does not degrade results quality. Images are represented with sparse semantic features that can be stored as an inverted index. Search complexity is drastically reduced by (1) considering the query feature dimensions independently and thus turning search into a concatenation operation and (2) pruning the index in function of a retrieval objective. To improve precision, the inverted index look-up is complemented with an exhaustive search over a fixed size list of intermediary results. We run experiments with three public collections and results show that our much faster method slightly outperforms an exhaustive search done with two competitive baselines.

1. Introduction

The development of social media and the democratization of digital cameras led to an increasingly important role of image based communication. Images were often a complement of textual data in the early days of the Web, while they now play the central role on platforms such as Instagram, Snapchat or Flickr and a growing one on others, such as Twitter. The increased importance of images is accompanied by a consequent research and development effort that aims at making large-scale image collections accessible. As part of this effort, content-based image retrieval (CBIR) tools are now standard components of search engines such as Google or Bing.

Increasingly powerful visual features were developed and exploited to improve CBIR accuracy. Prominent features that were proposed during the last decade include bags of visual words [24], Fisher vectors [20] and, more recently, convolutional neural network (CNN) features [15]. Retrieval scalability improvement focused on the following aspects (or a combination of them): (1) reduction of the size of feature vectors [13, 17]; (2) approximate search with partitioning trees [19] and (3) representation of the image collection with an inverted index structure [24, 10]. CBIR systems have to deal simultaneously with the accuracy of results and scalability of the retrieval process [13]. Finding an optimal trade-off between the two characteristics is a hard problem that we tackle here.

We introduce a CBIR framework that has (nearly) scale-free complexity and does not sacrifice accuracy. The main contribution is to consider query feature dimensions independently of one another. This modeling choice greatly simplifies the search process since it replaces complex mathematical operation by a concatenation. Semantic image features [16, 5] encapsulate significant information in each one of their dimensions and are used as main representation of image content. These features are sparse [10], a property that enables efficient representation of the image collection as an inverted index. To further improve accuracy, a reranking step is performed over a fixed size list resulting from the initial concatenation of results.

After a presentation of related work, we introduce the CBIR framework and analyze its components. We then evaluate the proposed method and, before concluding, discuss some of its limitations.

2. Related Work

The first line of relevant work concerns the creation of features that provide an accurate encoding of image content. During the last decade, the most widely used image retrieval features relied on the aggregation of local features, such as bags of visual words (BoVW) [24]. These approaches first extract local features, such as SIFT [18], and then aggregate them into a fixed size BoVW vector that describes the global properties of the image [24]. BoVW were improved through the introduction of higher-order image statistics in features such as *Fisher vectors* [20]. A problem shared by these descriptors is their high dimensionality and different compression methods were proposed to improve scalability.

In particular, [12] compressed Fisher vectors into a simpler representation named *VLAD* by using product quantization (*PQ*) [13]. With *VLAD+PQ* representation, 100 million image features are searched in approximately 250 ms on one core. While improving scalability, the aggressive compression performed by *VLAD+PQ* significantly decreases accuracy compared to the use of full *Fisher vectors*.

Convolutional Neural Networks (*CNN*) have superseded aggregated local features in image classification [7, 15] and retrieval [22, 10]. While accurate, CNN features usually have a size in the range of thousands of dimensions that makes their direct use for large-scale retrieval cumbersome. The compression of deep learning features has received increasing attention. One approach exploits Transductive SVMs and binary trees to create compact binary hashes [6]. The authors of [17] propose a method that learns binary descriptors in an unsupervised manner. The results obtained with compressed features are often close to those of full CNN vectors. However, the complexity of the search operation remains linear if an exhaustive search is performed.

The authors of [21] compared *CNN*, *VLAD* and *VLAD+PQ* in retrieval task on the YFCC100M collection that includes nearly 100 million images [25]. Results show that the accuracy of *CNN* features is roughly three times higher than that of *VLAD* and *VLAD+PQ*. Equally important, the paper shows that it is possible to reliably evaluate retrieval performance using an automatically created and imperfect ground truth.

A second line of work focuses on the development of semantic features such as Object Bank [16] or meta-classes [5] that exploit low-level or intermediate features in order shift image representations to a semantic space defined by the activations of an array of visual concept detectors. These authors aggregate multiple low-level features to learn the detectors and show that the resulting semantic features have higher accuracy than the basic features. A sparse variant of semantic features that was built on top of *CNNs* was introduced in [10]. This feature has only dozens of non-null dimensions and compares favorably with the basic *CNN* descriptors both in terms of accuracy and scalability. For instance, retrieving results in a collection of 100 million images takes hundreds of milliseconds if the collection is represented as an inverted index and stored in RAM.

Semantic features have also recently shown a considerable improvement of retrieval time for video search [14]. After concept detection, two approaches for concept adjustment are proposed. While the first one deals with the logical consistency among the concepts found in a video, the second one addresses their distributional consistency. These steps lead to a video representation consisting of 10 to 60 salient and consistent concepts. The authors report a retrieval time of 0.2 seconds on a single CPU core for a collection of 100 million videos. While much faster than raw *CNN* features or even *VLAD+PQ*, semantic features still require arithmetic operations to retrieve results and search complexity grows roughly linearly with the size of the collection.

A third line of relevant work concerns the efficient representation of image collections. Two main types of structures are used: partitioning trees [4] and inverted indexes [2]. Partitioning trees are well adapted for an approximate search over dense feature vectors and a number of variations of such structures are discussed in [19]. Classical kd-trees [4] are of limited use in high-dimensional spaces and approximations were proposed that implement either error bounds [1] or time bounds [3]. The authors of [19] perform a thorough evaluation of different types of tree structures and show that no structure performs best over all evaluation datasets. Depending of the dataset, best results are reported with randomized k-d trees and with a variant of a k-means tree. A distributed version of k-d trees is proposed in [19] in order to scale-up the search process. A 10^3 - 10^4 acceleration with a precision loss between 5% and 50% compared to exhaustive search is reported. However, the search time is still heavily dependent on the collection size and scaling-up the system for larger collections requires new machines.

Inverted indexes were first used in text retrieval since these documents have a sparse representation over the textual vocabulary [2] and they strongly reduce search time compared to forward indexes. They were then adapted to image retrieval when sparse features that encode image content efficiently became available [24]. Compared to decision trees, inverted indexes have the advantage of providing a better approximation of exhaustive search if very frequent dimensions (i.e. similar to stop words in text retrieval [2]) or very rare dimensions are removed. The results reported in [10, 14] show that the use of an inverted index for image and video retrieval improves search time by several orders of magnitude. However, even if they are applied to only dozens of non-null dimensions encoded by the inverted index, arithmetic operations are still needed and search time increases with the size of the collection.

3. Retrieval Framework

We can state our general objective as: ***Given an image collection C and a query image q, a set of x similar results should be accurate and retrieved in a time that is nearly independent of collection size.*** In Figure 1, we illustrate the retrieval framework introduced here to tackle this objective.

There are two main steps in the pipeline that aim: (1) to create an intermediary list of results and (2) to rerank the elements of this list respectively. Our main contribution is to propose a method that retrieves the intermediary list in a time that is independent of the collection scale. If x is the retrieval objective (i.e. the number of images to be retrieved by the retrieval system), in practice, the size of

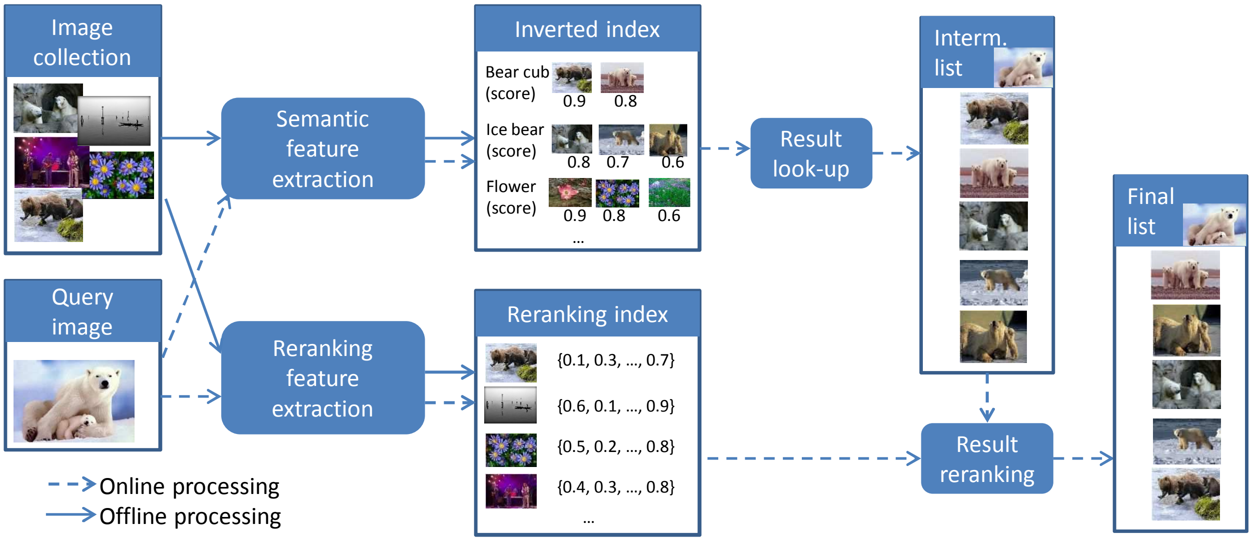

Figure 1. Illustration the proposed scale-free CBIR pipeline. The data and results are represented in rectangles and the different processing steps performed over data are presented in rounded rectangles. The collection indexing process is represented by full lines and the search process by dotted lines.

the intermediary list of results is given by a linear function $f(x) = k \times x$, with $k \geq 1$. The choice of a list larger than x is motivated by the fact that relevant results might be found beyond x depth and could be surfaced during the reranking that provides the final list of results. Departing from existing practices, the look-up process considers the dimensions of the query representation independently from one another. First, in order to obtain relevant results, it is crucial that each dimension of the features used for look-up encodes as much information as possible. Second, these features should be generic enough in order to accommodate the strongly diversified types of content available in social media image collections. Third, these features should be sparse in order to obtain a compact inverted index that can be easily stored in RAM for faster processing.

Among existing image descriptors, semantic features fulfill the three requirements since their dimensions are activations of semantically meaningful visual concepts, they can incorporate tens of thousands of different concepts [5] and have optimal performance in a sparse form [10]. This choice simplifies the exploitation of the inverted index and makes it independent of the collection scale, as detailed in Subsection 3.3. For instance, assuming that the query image is described by *bear cub* and *ice bear*, the intermediary list from Figure 1 contains a concatenation of the inverted index entries associated to these concepts. This concatenation is needed in order to make sure that the retrieval objective is met. In our example, if $f(x) = 5$ images are needed and if *bear cub* and *ice bear* inverted index entries have 2 and 3 associated images, both concepts should be used.

Each dimension of the query and collection images encodes the likelihood of a visual concept but an optimal similarity could result from an intersection of several concepts. To deal with this problem, we introduce a result reranking step that refines the intermediary list to refine the results of the look-up process. This might be necessary in order to cope with situations in which image similarity is best defined by a combination of concepts. For instance, the final results in figure 1 favor images that are associated to both *bear cub* and *ice bear*, while the temporary list has a top result associated to *bear cub* but not to *ice bear*. While semantic features are needed to simplify look-up, reranking can be executed with any feature since it only acts on a fixed size list of intermediary results. The main criterion for the choice of reranking features should be their accuracy.

The retrieval pipeline includes two main processing flows, dedicated to the collection and to the query. As usual, collection processing (illustrated by the continuous arrows in Figure 1) is performed offline. In the implementation presented in 1, semantic features and low-level features are first extracted for each image of the collection. Then these features are stored in the inverted and reranking indexes. Note that if reranking is also based on semantic features, a single feature extraction step is needed. We evaluate the two combinations of features in the experimental section.

Existing CBIR pipelines fail to search results independently of the collection size. This is explained by the fact that they implement a result ranking that is based on image features whose dimensions are exploited together. This classical similarity computation entails a number of arithmetic operations that have a significant computational cost. Costly operations, such as multiplications, are applied to *all* non-null dimensions of the query to compute the ranking scores for *all* the corresponding collection documents. Under a dimension independence hypothesis for the query image, classical arithmetic operations are replaced by a much lighter look-up process.

3.1. Semantic Features

Semantic features encode the content of an image using the activations of an array of n visual concept classifiers [5, 16] and can be written as $D = \{(v_1, s_1), (v_2, s_2), ..., (v_n, s_n)\}$, with v_j the j^{th} visual concept and s_j its activation score. This representation is applicable to any type of low-level or intermediate visual features. A sparse version of semantic features was introduced in [10] and is discussed in detail there. In this variant, only a small fraction c_i of scores s_i are non-zero (i.e. $c \ll n$). This representation is noted D_c and it corresponds to the intuition that only a limited number of concepts are actually useful to represent an image. Extensive evaluation done on datasets surch as Wikipedia Retrieval, MIR Flickr and NUS-WIDE in [11] results show that sparse features' performance is quite stable for $c \geq 10$. Sparsity is a desirable property here since it indicates that semantic features encode a large quantity of information on a small number of dimensions. We follow the authors of [10] and implement a semantic feature to support scale-free retrieval. Visual models v_i are created for the $n = 17,462$ ImageNet concepts that have at least 100 representative images. These models are learned independently of one another, using negative examples from a diversified negative set that is sampled from ImageNet concepts that are not included in the feature. The learning of independent SVMs for visual concepts introduces introduces a supplementary step compared to the direct use of CNN features. However, it has the advantage of allowing fast enrichment of the semantic representation. For instance, if a CNN is used directly and one wants to add a new concept, a full retraining of the model is needed. With independent SVMs, a single new concept is learned in near real time. This property is especially useful in dynamic environments, such as social media, where new concepts are continuously created and need to be represented. To scale-up both training and test, models are learned with the default linear SVM model from liblinear [9].

We illustrate the sparse representation of 3 images in Figure 2 using $c_i = 5$ concepts. Although imperfect, the semantic features capture the most important concepts of the images. For instance, the image to the left of the figure is correctly annotated with *gondola*, *punter* and *gondolier*. The association of *sampan* is incorrect but explained by the fact that this concept is visually similar to *gondola*. In ImageNet, *raceway* is defined as *a canal for a current of water* and is thus one of the common contexts of *gondola*. The only concept that is wrongly associated to the image in the middle of Figure 2 is *guitarist*. However, this concept is still related to the general theme of the image. All concepts attributed to the rightmost image are relevant, with the most informative ones being *bear cub* and *ice bear*.

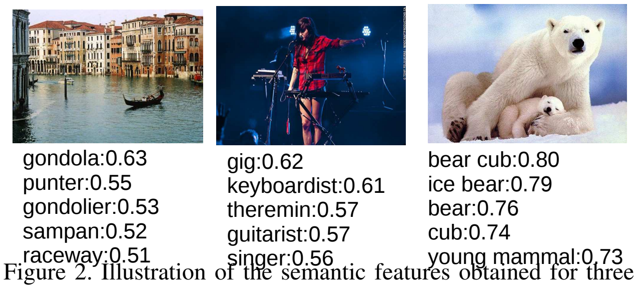

Figure 2. Illustration of the semantic features obtained for three images with $c_i = 5$. Top visual concepts are presented along with their classification scores.

3.2. Query Image Representation

Query images have two representations, corresponding to the described look-up and reranking steps. Look-up is done with semantic features, while reranking exploits the same features that were stored in the reranking index.

3.2.1 Query Representation Analysis

We depart from existing CBIR systems through the query representation exploited during the first retrieval step performed over the inverted index. First, the dimensions of query features are considered independently of one another during retrieval. This property is essential since it enables an inverted index querying that replaces classical arithmetic operations [2] with a concatenation of inverted index entries. Second, since the retrieval process has a predefined objective to find x images, the query and image collections representations are different even if both are rooted in the same semantic space. Depending on the query image, a variable number of query dimensions are needed to fulfill the retrieval objective, while the collection images are represented with a fixed c_i number of non-zero dimensions.

3.2.2 Query Representation Implementation

As we mentioned, a core requirement for a successful implementation of the proposed retrieval pipeline is that each dimension of image features encodes as much information as possible. Naturally, the query representation exploits a version of D in which visual concepts are ranked according to their activation score. This ranking gives priority to visual concepts v_i that are most salient in the query.

3.3. Collection Representation and Querying

As we mentioned, the proposed retrieval pipeline includes two retrieval steps. The first one exploits an inverted index structure to reduce the search space to an intermediary list. The list can be further processed in real-time during the second retrieval using image representations from the reranking index using a forward search.

When image features are sparse, as it is the case for D_c, a forward index would contain mostly zero values that do not contribute to the similarity between images. The collection can then be efficiently structured as an inverted index I_I that includes, for each dimension of the features only non-zero values. Similar to collection features, the query descriptor D_q contains c_q non-zero dimensions that are the only one used for querying I_I. A similarity measure is computed to retrieve the closest neighbors of the query. This operation is much more efficient than a forward search since it ignores the all zero values of sparse features. It is however dependent of the collection size since index entries will be richer for larger collections and more mathematical operations are needed to compute similarities.

To our knowledge, existing inverted index implementations dedicated to CBIR [24, 10] consider feature dimensions jointly to compute image similarities. Consequently, in order to obtain results that are equivalent to an exhaustive forward search, all images associated to inverted index entries need to be saved, thus enlarging its total size. In contrast, following our feature dimension independence hypothesis, we drastically reduce the number of images per entry of I_I by relating it to the retrieval objective through $f(x)$ and storing only the most salient images for each visual concept. This representation of the collection is independent of the its size since I_I will contain at most $f(x)$ images for each of the n dimensions stored in it. More importantly, the costly mathematical operations needed to compute similarities are replaced by a concatenation of I_I entries that are associated to decreasingly important dimensions of the query representation D_q. The iteration over several query dimensions is needed since not all index entries include a sufficient number of images to fulfill the retrieval objective. The only supplementary operation during concatenation of results is the removal of duplicate image identifiers that can appear when $c > 1$ in D_c (i.e. the same collection image being associated to more than one concept v_i of the query).

3.4. Time Complexity Analysis

The search process from figure 1 entails three main steps whose algorithmic complexity is discussed hereafter:

- **query indexing** is independent of the test collection size and, as a consequence, has a $\mathcal{O}(1)$ complexity.
- **intermediary results look-up** is a simple concatenation operation over an inverted index. The number of entries of the index is independent of the test collection size since images are indexed with a fixed-size visual semantic feature. The complexity of the look-up process only depends of $f(x)$, which is itself independent of the total collection size and is run in $\mathcal{O}(1)$. If duplicate images appear, they are removed from the list of results but the complexity of this step is much smaller than that of the result reranking which is later operated over $f(x)$.
- **result reranking** involves the comparison of the reranking features of the query image to those of the top $f(x)$ images from in the intermediary list of results. $f(x)$ being independent of the collection size, the complexity of similar image search is itself independent of the scale of the entire collection ($\mathcal{O}(f(x))$). For instance, is $f(x) = 10,000$, the cost of the reranking itself is the same regardless if the collection size is 1, 10 or 100 million images. The only scale dependent operations are the extraction of $f(x)$ features from the reranking index. This index is stored as a database table, which is indexed by the image identifiers and the physical access to content is done in $\mathcal{O}(\log n)$. However, similar image search during reranking step involves arithmetic operations and is much more complex than the access itself. In practice, the time needed for similarity computation significantly exceeds the one needed for access to content in the database and the complexity of the reranking step can itself be considered as nearly scale-free.

4. Evaluation

We evaluate the performance of the proposed CBIR pipeline using three public datasets. An ad-hoc retrieval scenario is retained, i.e. a wide array of queries that are not known in advance can be presented to the system. Mean average precision (mAP) provides a robust estimation of system performance [26] and is adopted here. The scale collections whose scale ranges from small to very large. We introduce the datasets, then evaluate overall performance and finally vary important parameters of the pipeline. The scale collections whose scale ranges from small to very large.

4.1. Evaluation datasets

Pascal VOC 2007 [8] (**VOC07** hereafter) is a sample of 9,963 Flickr images. It includes a complete assessment of the presence of 20 concepts in the collection images. A split that includes 1,000 test and 8,963 collection images is created and will be published to facilitate reproducibility.

Wikipedia Retrieval 2010 [26] (**Wiki** hereafter) was created as part of the ImageCLEF evaluation campaign[1]

[1] http://www.imageclef.org/

and includes 237,434 Wikimedia images. The collection includes a wide range of content and it is thus fitted for ad-hoc image retrieval experiments since diversified queries can be launched over it. We exploit the 2010 campaign query set and ground truth that include 118 query images associated to 70 diversified topics.

Yahoo Flickr Creative Commons 100M [25] (abbreviated **YFCC**) is currently the largest publicly available multimedia collection. It includes a total of 99.2 million images licensed under different versions of Creative Commons licenses. The images were collected between 2004 and 2014 from a significant subset of Flickr users and the collection can thus be considered a representative sample of the Web corpus. A mirror of the dataset is not yet available and we collected the 96.7 million still images that were still available in May 2015. A subset of 50 diversified textual topics from ImageCLEF Wikipedia retrieval 2010 and 2011 query sets is selected. Three image queries per topic are used in the evaluation, resulting in a total of 150 examples. Although a manually created ground truth is not available for YFCC, it has been showed that reliable evaluation can be performed with an automatically created ground truth [21]. It is created by considering the tags associated to images by the users. The only inconvenient of this approach is that performance measures are underestimated by a factor of 3 to 4 due to the incomplete tagging of the collection.

Following the usual TREC evaluation protocol, mAP scores are computed over the top 1,000 results obtained for all queries, corresponding to a retrieval objective $x = 1,000$ [26] for WIKI and YFCC. Given the small total size of VOC07, mAP@100 is used in this case.

4.2. Retrieval pipeline implementations

As we mentioned, the scale-free retrieval pipeline includes two steps: (1) looking-up an intermediary list of results whose size is $f(x)$ and (2) refinement of results through an exhaustive search over the content of the intermediary list. Tests are run with the first step only and with the full pipeline.

The following image features are extracted to implement our pipeline and to create baseline CBIR systems:

1. Extraction of VGG raw features from the standard ImageNet model with $1,000$ concepts provided in [23]. An L^2 normalized version of the last fully connected layer ($fc7$), which consists of $4,096$ dimensions, is exploited here. This layer is kept because its activations encode an intermediate representation of image content. VGG was shown to be highly effective for both image classification [23] and retrieval [21]. An exhaustive search with VGG over each test collection constitutes a first strong baseline here.

2. Computation of sparse semantic features, noted SEM, using an improved version of the ones introduced by the authors of [10]. Notably, a ratio of $1 : 100$ between positive and negative examples was empirically determined as optimal and is used here instead of a fixed size negative class described in [10]. This ratio was empirically obtained after a grid search with values between 1 and 500. These features are computed using the $fc7$ layer of VGG as basic feature. Collection images are represented by a predefined number of dimensions c_i in the inverted index. Query images are represented by a variable number of dimensions necessary to fulfill the retrieval objective x. These semantic features were shown to have competitive CBIR performance when compared to the basic CNN features from which they are extracted. They constitute a second strong baseline for our experiments. If the reranking index is also based on semantic features, a fixed number c_r are used.

	Dataset		
	VOC07	WIKI	YFCC
	mAP[%]		
VGG	55.28	16.83	4.95
SEM	60.37	19.55	4.41
INT	55.64	16.13	3.69
REF_{VGG}	56.22	18.73	**5.35**
REF_{SEM}	**60.59**	**19.56**	5.04

Table 1. CBIR performance obtained with VGG and SEM, two strong baselines, and instantiations of the proposed retrieval pipeline. Reported performance is the best obtained through a grid search of c_i, the number of top concepts retained for each image in the inverted index. Results are reported for $c_i = 20$, $c_i = 3$ for VOC07 and WIKI. For YFCC, results are obtained with $c_i = 20$ and $c_i = 5$ for REF_{VGG} and REF_{SEM}. The value of $f(x)$, which determines the reduction of search complexity, is $1,000$, $1,000$ and $10,000$. REF_{SEM} is run $c_r = 20$ dimensions for the construction of the R_I in all configurations.

We test the following variants of our pipeline:

- INT - intermediary list of results obtained using only the look-up of the inverted index from in figure 1.
- REF_{VGG} - refined list of results obtained after reranking of the intermediary results with VGG features.
- REF_{SEM} - refined list of results obtained after reranking of the intermediary results with semantic features.

4.3. Overall Results

In Table 1, we present the results obtained for the three test collections. REF_{VGG} and REF_{SEM}, the two implementations of the proposed pipeline have interesting performance compared to their associated baselines VGG and

	YFCC (mAP[%]) / c_i				
	1	5	10	15	20
INT	3.88	3.72	3.69	3.69	3.69
REF_{VGG}	4.95	5.2	5.31	5.34	5.35
REF_{SEM}	4.59	5.04	5.04	5.01	5.02

Table 2. CBIR performance for different values of c_i, the number of top concepts in I_i. The value of the objective function is $f(x) = 10,000$. REF_{SEM} is run $c_r = 20$ dimensions for the construction of R_I in all configurations.

	YFCC (mAP[%])		
	f(x)		
	$1,000$	$10,000$	$100,000$
INT	3.69	3.72	3.69
REF_{VGG}	4.2	5.35	4.88
REF_{SEM}	4.72	5.02	4.51

Table 3. Performance for different $f(x)$, the value of the objective function. I_I is built using $c_r = 20$ for REF_{SEM}.

SEM. Very interestingly, the largest performance gains (8.1% and 14.3% compared to VGG and SEM) are obtained for YFCC, the dataset which best approximates the Web corpus here. The performance gain for YFCC also entails a drastic reduction of search complexitysince only $10,000$ images out of 96.5 million are used for the reranking step. For VOC07 and WIKI, our best performance obtained is roughly equal to that of SEM, the best baseline.

Performance is still interesting but lower than that of the baselines even with a simple usage of look-up (INT). In this case, the gap is growing from the smallest to the largest dataset. Another interesting finding is that the baselines behave differently for YFCC compared to VOC07 and WIKI. VGG has higher performance at large scale (YFCC), while SEM is consistently better for the other datasets. Following a similar trend, the best overall results are obtained with REF_{VGG} for YFCC and REF_{SEM} for VOC07 and WIKI.

4.4. Parameter Evaluation

In addition to the overall evaluation, we test the robustness of the proposed pipeline by varying its main parameters on the YFCC collection. First, we set $f(x) = 10,000$ for reranking, and report results with different values of c_i in table 2. mAP scores are rather stable for the three but, interestingly, they do not behave identically. While the best score for INT is obtained for $c_i = 1$, the maximum value for REF_{VGG} is obtained for $c_i = 20$ and that for REF_{SEM} for $c_i = 5$. Beyond these values, the quality of the results starts to decrease slowly and we did not inserted results. This behavior confirms the conclusions of [10] regarding the fact that an optimal semantic feature includes a few dozens of non-zero dimensions for each image. The better behavior of INT for $c_i = 1$ is probably explained by the fact that, when only look-up is used, the top results should be as clean as possible at the top of the list. Inversely, the reranking step manages to surface good quality results that are scattered across the intermediary list.

$f(x)$, the size of the intermediary list of results, approximates the scalability gain obtained with our approach. We set $c_i = 20$ and test with $f(x) = \{1,000, 10,000, 100,000\}$, corresponding to scalability gains of six, five and four orders of magnitude respectively. The obtained results for the three configurations from table 3. The look-up of I_I is efficient since the performance of REF_{VGG} and REF_{SEM} decreases beyond $f(x) = 10,000$. Otherwise said, best performance is obtained with only 10^{-4} of the full YFCC collection used for reranking. This finding indicates that most relevant results are placed near the top of the intermediary results list and an efficient reranking can be deployed to refine results.

4.5. Scalability Evaluation

Following the complexity analysis presented in Subsection 3.4, we evaluate the search time for the REF_{SEM} with that of baselines SEM and VGG. Below, we exclude the query image processing since it is independent of the tested collection size when comparing the different methods. The extraction of VGG features takes approximately $20ms$ on a Titan X and that of semantic features another $25ms$.

SEM search uses an inverted index that is stored in RAM and the search process implemented in C++. VGG uses a forward index that is also stored in RAM and the exhaustive search process is implemented in C++, using SSE2 instructions to accelerate computation. To ensure comparability, the sparsity of semantic features is $c_i = 20$ for SEM and REF_{SEM}. The intermediary results lists has a size $f(x) = 10,000$ for REF_{SEM}. YFCC samples of 1 and 10 million images and the full collection are tested to assess the variation of search time for the three methods.

For the full collection, the results from table 4 indicate that search is roughly one and three orders of magnitude faster for REF_{SEM} compared to SEM and VGG. Equally important, the memory footprint is much smaller for REF_{SEM} since only the pruned index is stored in RAM while the full inverted index and the full forward indexes are stored in RAM for SEM and VGG respectively. That search time increases linearly for SEM and VGG, while it only increases by 2.8% and 5.6% when increasing the collection size from 1M to 10M and from 10M to 96.5M images for REF_{SEM}. As expected, VGG has the worst behavior since it implies as brute force search operation over $4,096$ dimensions per image. SEM is better than REF_{SEM} for 1M images while its performance is worse for 10M and the full scale collection.

The search time results empirically confirm the complexity analysis presented in Subsection 3.4. They show that

	Collection size		
Method	1M	10M	96.5M
REF_{SEM}	107 ms	110 ms	113 ms
SEM	19 ms	192 ms	1935 ms
VGG	2034 ms	20456 ms	204634 ms

Table 4. Reranking search time for different collection sizes as an average of 100 queries. The presented search times include only the search steps for each method. Experiments are run on a single core using an INTEL Xeon E5-2643 at 3.3GHz.

search time for REF_{SEM} increases only marginally with the size of the collection. This variation is only due to the logarithmic complexity of the access to the features needed for reranking.

5. Discussion and conclusions

Our approach to (nearly) scale-free image retrieval has interesting advantages but also a number of limitations:

- While large, the array of concepts used in the implementations of the semantic features offers an incomplete coverage of concepts that are depicted in Web images. However, this risk also appears for lower-level features, including VGG that was exploited to build the semantic features. This limitation can be tackled via the extension of ImageNet with manual labeling or, in a more scalable manner, via the exploitation of noisy Web corpora to learn more classifiers [10].

- A pruned inverted index is exploited to speed-up the search process and not all the images of large scale collection are accessible. Overall, the diversity of the obtained results could be smaller compared to that of exhaustive search processes that consider the entire collection. However, diversity can be controlled by diversifying the query concepts that are exploited to populate the intermediary list of results and/or by increasing the retrieval objective.

- The semantic features are learned with simple linear classifiers whose performance is probably lower than that of more complex methods. This choice is deliberate since in ensures scalability for both the training of semantic features and their usage.

- Assuming that the query image is part of the collection, our pipeline does not guarantee that this image will be in result list. This happens in cases when the scores of the query's top concept(s) do not place it among the images retained in the inverted index I_I. This limitation can be circumvented by relaxing the I_I pruning in order to include more images per dimension and by looking-up results close to the activation scores of the query image for each dimension.

We introduced a pipeline that drastically reduces the complexity of the CBIR process and does not degrade the quality of results compared to exhaustive search. We showed it is indeed possible to implement a nearly scale-free CBIR, provided that appropriate answers are proposed to focused research questions that were tackled. To reduced the search process complexity, queries feature dimensions are exploited independently of one another and the test collection is efficiently stored as a pruned inverted index. Our results indicate sparsified semantic features can be exploited to represent images and to optimize the quantity of image-related information encoded by each dimension. After this reduction, any feature can be used to refine a fixed-size intermediary list of results and improve the overall performance of the system. The refinement step is nearly independent of the collection scale since only the retrieval of features that correspond to the intermediary list depends on the collection with logarithmic complexity.

Future work will focus on improving different aspects of the proposed approach. First, if we include query processing, the overall retrieval time for 96.5 million images is just over 150 ms. We will work towards further reducing it. The computation of semantic features takes $25ms$ and an approximate version of it will be implemented in order to speed it up. Equally important, the implementation of the reranking index can be further optimized, for instance by exploiting an efficient NoSQL representation format. Second, the pipeline is generic enough to easily exploit better reranking features that will be incorporated upon availability. Third, one current limitation regards the coverage of the semantic feature. We will investigate ways to add supplementary visual concept detectors so as to cover a larger spectrum of queries for instance by using webly supervised methods to add new concepts [10].

References

[1] S. Arya, D. M. Mount, N. S. Netanyahu, R. Silverman, and A. Y. Wu. An optimal algorithm for approximate nearest neighbor searching fixed dimensions. *J. ACM*, 45(6):891–923, Nov. 1998. 2

[2] R. A. Baeza-Yates and B. Ribeiro-Neto. *Modern Information Retrieval*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1999. 2, 4

[3] J. S. Beis and D. G. Lowe. Shape indexing using approximate nearest-neighbour search in high-dimensional spaces. In *Proceedings of the 1997 Conference on Computer Vision and Pattern Recognition (CVPR '97)*, CVPR '97, pages 1000–, Washington, DC, USA, 1997. IEEE Computer Society. 2

[4] J. L. Bentley. Multidimensional binary search trees used for associative searching. *Commun. ACM*, 18(9):509–517, Sept. 1975. 2

[5] A. Bergamo and L. Torresani. Meta-class features for large-scale object categorization on a budget. In *Computer Vision and Pattern Recognition*, 2012. 1, 2, 3, 4

[6] H. Cevikalp, M. Elmas, and S. Özkan. Towards category based large-scale image retrieval using transductive support vector machines. In *Computer Vision - ECCV 2016 Workshops - Amsterdam, The Netherlands, October 8-10 and 15-16, 2016, Proceedings, Part I*, pages 621–637, 2016. 2

[7] D. Ciresan, U. Meier, and J. Schmidhuber. Multi-column deep neural networks for image classification. In *CVPR 2012*, pages 3642–3649, 2012. 2

[8] M. Everingham, L. Van Gool, C. K. Williams, J. Winn, and A. Zisserman. The pascal visual object classes (voc) challenge. *International journal of computer vision*, 88(2):303–338, 2010. 5

[9] R.-E. Fan, K.-W. Chang, C.-J. Hsieh, X.-R. Wang, and C.-J. Lin. LIBLINEAR: A library for large linear classification. *Journal of Machine Learning Research*, 9:1871–1874, 2008. 4

[10] A. Gînsca, A. Popescu, H. L. Borgne, N. Ballas, P. Vo, and I. Kanellos. Large-scale image mining with flickr groups. In *Proc. of Multimedia Modelling Conf. 2015*, 2015. 1, 2, 3, 4, 5, 6, 7, 8

[11] A. L. Ginsca. *Leveraging large scale Web data for image retrieval and user credibility estimation*. Theses, Télécom Bretagne ; Université de Bretagne Occidentale, Nov. 2015. 4

[12] H. Jégou, M. Douze, C. Schmid, and P. Pérez. Aggregating local descriptors into a compact image representation. In *CVPR 2010*. IEEE Computer Society. 2

[13] H. Jegou, F. Perronnin, M. Douze, J. Sanchez, P. Perez, and C. Schmid. Aggregating local image descriptors into compact codes. *IEEE PAMI*, 34(9):1704–1716, 2012. 1, 2

[14] L. Jiang, S.-I. Yu, D. Meng, Y. Yang, T. Mitamura, and A. G. Hauptmann. Fast and accurate content-based semantic search in 100m internet videos. In *Proceedings of the 23rd Annual ACM Conference on Multimedia Conference*, pages 49–58. ACM, 2015. 2

[15] A. Krizhevsky, I. Sutskever, and G. Hinton. Imagenet classification with deep convolutional neural networks. In *Adv. in Neural Information Processing Systems*, 2012. 1, 2

[16] L.-J. Li and al. Object bank: A high-level image representation for scene classification & semantic feature sparsification. In *NIPS*, 2010. 1, 2, 4

[17] K. Lin, J. Lu, C. Chen, and J. Zhou. Learning compact binary descriptors with unsupervised deep neural networks. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*, pages 1183–1192, 2016. 1, 2

[18] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, 60(2):91–110, 2004. 1

[19] M. Muja and D. G. Lowe. Scalable nearest neighbor algorithms for high dimensional data. *IEEE Trans. Pattern Anal. Mach. Intell.*, 36(11):2227–2240, 2014. 1, 2

[20] F. Perronnin, Y. Liu, J. Sánchez, and H. Poirier. Large-scale image retrieval with compressed fisher vectors. In *CVPR*, 2010. 1

[21] A. Popescu, E. Spyromitros-Xoufis, S. Papadopoulos, H. Le Borgne, and I. Kompatsiaris. Towards an automatic evaluation of retrieval performance with large scale image collections. In *MMCommons 2015 workshop*. 2, 6

[22] A. S. Razavian, H. Azizpour, J. Sullivan, and S. Carlsson. Cnn features off-the-shelf: an astounding baseline for recognition. *CoRR*, abs/1403.6382, 2014. 2

[23] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *CoRR*, 2014. 6

[24] J. Sivic and A. Zisserman. Video google: A text retrieval approach to object matching in videos. In *ICCV 2003*. 1, 2, 5

[25] B. Thomee, D. A. Shamma, G. Friedland, B. Elizalde, K. Ni, D. Poland, D. Borth, and L. Li. The new data and new challenges in multimedia research. *CoRR*, abs/1503.01817, 2015. 2, 6

[26] T. Tsikrika, J. Kludas, and A. Popescu. Building reliable and reusable test collections for image retrieval: The wikipedia task at imageclef. *IEEE MultiMedia*, 19(3):24–33, 2012. 5, 6