A Group Testing Framework for Similarity Search in High-dimensional Spaces

Miaojing Shi, Teddy Furon, Hervé Jégou

A Group Testing Framework for Similarity Search in High-dimensional Spaces

Miaojing Shi*
Peking University

Teddy Furon
Inria

Hervé Jégou
Inria

ABSTRACT

This paper introduces a group testing framework for detecting large similarities between high-dimensional vectors, such as descriptors used in state-of-the-art description of multimedia documents. At the crossroad of multimedia information retrieval and signal processing, we produce a set of group representations that jointly encode several vectors into a single one, in the spirit of group testing approaches. By comparing a query vector to several of these intermediate representations, we screen the large values taken by the similarities between the query and all the vectors, at a fraction of the cost of exhaustive similarity calculation.

Unlike concurrent indexing methods that suffer from the curse of dimensionality, our method exploits the properties of high-dimensional spaces. It therefore complements other strategies for approximate nearest neighbor search.

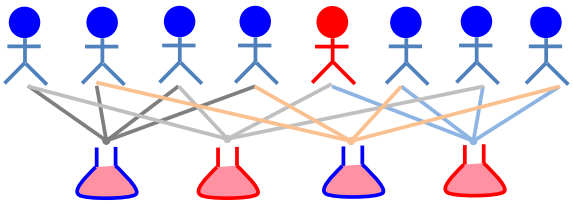
Our preliminary experiments demonstrate the potential of group testing for searching large databases of multimedia objects represented by vectors. We obtain a large improvement in terms of the theoretical complexity, at the cost of a small or negligible decrease of accuracy. We hope that this preliminary work will pave the way to subsequent works for multimedia retrieval with limited resources.

1. INTRODUCTION

With the apparition and massive growth of social networks, and simultaneously the massive proliferation of personal capture devices such as smartphones, research in multimedia has entered the age of "big data". In this context, the scale to be considered by search engines is in the order of billions of heterogeneous documents: mixtures of image, sound, video, text, or even more complex data such as links of various nature in social networks.

This raises the issue of resources required to search and organize such large collections. Both efficiency and memory are critical issues. The need for efficient search has maintained a sustained effort of research in various communities, leading to some major improvements in the last decades. One of the most noticeable advances is arguably the introduction of Locality-sensitive hashing [7] (LSH). This indexing strategy and its derivatives, such as E2LSH [4], were the first to exhibit provable sub-linear search in arbitrary spaces. However, this method trades memory for efficiency, and the practical complexity grows dramatically with the dimensionality. For descriptors of intermediate dimensionality such as SIFT descriptors [15], many hash functions are required and the indexing structure has a prohibitive memory cost. Data-aware methods like FLANN [20] or LSH variants employing the multiple-probe mechanisms [18] are preferred in practice. The situation is even worse when considering the high-dimensional vectors used for image retrieval, such as Fisher vectors [22] or bag-of-words [24]. To our knowledge, all sub-linear indexing strategies are ineffective for such vectors, and not used in state-of-the-art image search engines.

Group testing was born during World War II [6]. Facing an increasing number of injuries, the US army decided to make blood donation mandatory for soldier recruits. However, a fraction of them were infected by syphilis. Unfortunately, syphilis testing back then was expensive. Realizing that there is no point running an exhaustive testing procedure over a large population if a small fraction of individuals are infected, some blood samples were pooled into one mixture and tested against syphilis. If the test was positive, the blood samples were individually tested.



This very promising idea was not actually put to use. The seminal paper on it dates back to 1943 due to Robert Dorfman [5]. Nowadays, group testing finds application in pharmacology, bioinformatics, genome screening, etc. This paper considers the potential of this approach for similarity search, a key ingredient of multimedia systems.

Figure 1: Group testing: An historical motivation.

*Miaojing Shi has worked on this paper while he was a visiting student at Inria Rennes.

Preprint version.
MM'14, November 3–7, 2014, Orlando, Florida, USA.

Sketching is another class of approximate search commonly used for high-dimensional vectors. Sketches in Hamming space, also derived from LSH, are popular [3, 17, 25]. In this case, each vector is mapped to a compact code, typically a binary vector, in order to reduce the complexity by a constant factor. The search is exhaustive, meaning that the code associated with the query is compared to all the database codes. The speed-up comes from the fact that it is faster to compare short binary vectors than long floating-point vectors, typically saving one to two orders of magnitude in terms of search efficiency. This approach is supported by theoretical results [3, 2] showing that the Hamming distance between binary vectors provides an approximation of the cosine similarity between the original Euclidean vectors. Note that this approach is also termed LSH in the recent literature. However, we stress that this approach does not offer the sub-linear complexity underlying the first LSH algorithms [7]. Recent developments in this line of research include compressed-domain distance estimation based on more general codes, such as product quantization [11].

In this paper, in a complementary manner, we are interested in reducing the number of measurements by avoiding the exhaustive comparison of the query with all the documents. The motivation is therefore in the spirit of the recent super-image approach [16]. Yet our approach is not specific to a particular image search setup and only requires that the indexed vectors satisfy some properties: being truly high-dimensional, with zero mean and unit norm, and compared with cosine similarity[1]. These assumptions are satisfied by state-of-the-art descriptors such as the Fisher vector [22], VLAD [12] or more recently the T-embedding [13].

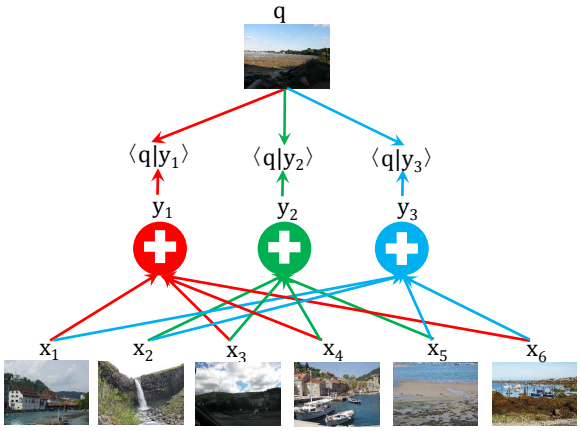
In other terms, the framework proposed in this paper is an attempt to explicitly exploit the properties underlying high-dimensional spaces in a search mechanism. More precisely, we consider a group testing approach as illustrated in Figure 1. In our similarity search scenario, for a given query we aim at detecting the largest similarities from a limited number of measurements, typically one order of magnitude smaller than what is formally required with an exhaustive comparison.

Our framework is formally introduced in Section 2, where we detail how we form overlapping groups of images, with each group being represented by a simple vector. Section 3 considers several score estimation strategies relying on pseudo-inverse reconstruction, sparse decoding and group testing methods. Section 4 presents a simple mathematical model to stress an important phenomenon known as dilution in group testing. The potential of our approach is evaluated in the experimental Section 5. Section 6 concludes the paper.

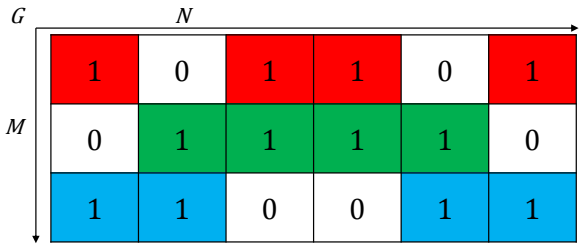
2. THE GROUP TESTING FRAMEWORK

Let us consider a collection of images described by vectors $\mathcal{X} = \{x_1, \ldots, x_j, \ldots, x_N\}$, $x_j \in \mathbb{R}^D$, and a D-dimensional query vector $q \in \mathbb{R}^D$ representing the query image. The images are compared with cosine similarity. We denote by $u_j = \langle q|x_j \rangle = q^\top x_j$ the similarity between the query and the j^{th} database image.

[1]The three last assumptions can be obtained by performing PCA and re-normalizing the resulting vector. This modifies the comparison metric but often has little impact of the performance for the application.



(a) Group representation



(b) Group matrix

Figure 2: Top: an illustration of the proposed group testing framework for image search. y_i denotes the i^{th} group vector, which is the sum of the image vectors of current group, e.g., $y_1 = x_1 + x_3 + x_4 + x_6$; $\langle q|y_i \rangle$ corresponds to the cosine similarity between group vector y_i and query vector q. Bottom: Group matrix G with its element g_{ij} being one if the j^{th} image belongs to the i^{th} group, otherwise $g_{ij} = 0$.

The similarities between a given query and the entire collection are therefore obtained by

$$\mathbf{u} = [u_1, \ldots, u_N]^\top = X^\top q, \tag{1}$$

where X is a $D \times N$ matrix with each column being the corresponding image vector x_j in database.

To avoid this costly matrix-vector operation, we consider the group testing framework illustrated in Figure 2. The image database vectors $x_1, ..., x_N$ are gathered into M overlapping groups $\mathcal{H}_1, \ldots, \mathcal{H}_M$. Each group $\mathcal{H}_i$ is described by a D-dimensional vector y_i, which is the sum of its image vectors:

$$y_i = \sum_{j \in \mathcal{H}_i} x_j. \tag{2}$$

Formally, the membership of an image to a group is given by a sparse $M \times N$ matrix $G = [g_{ij}]$, such that $g_{ij} = 1$ if image x_j belongs to group $\mathcal{H}_i$, otherwise $g_{ij} = 0$. Therefore, the group representation is obtained as

$$Y = X \cdot G^\top. \tag{3}$$

For a given query image q, our objective is to avoid the exhaustive comparison (1) of the query with the N images in the database. Instead, we calculate its similarities with the M group vectors Y as

$$\mathbf{v} = [v_1, ..., v_M]^\top = Y^\top q, \tag{4}$$

The group similarity vector $\mathbf{v}$ is M-dimensional, and is related to the image similarity vector by

$$\mathbf{v} = GX^\top q = G\mathbf{u}. \tag{5}$$

The objective is to infer the similarity values $\mathbf{u}$ from the measurements $\mathbf{v}$. Equation (5) is underdetermined because $M < N$, meaning that it has infinite number of solutions. In other terms, without any further assumptions, it is not possible to recover $\mathbf{u}$ from $\mathbf{v}$.

However, there are many ways to single out a solution. For instance, the standard approach described in Section 3.1 relies on the pseudo-inverse.

Sparse recovery [14] is another approach, where the typical assumption is that $\mathbf{u}$ is sparse: over N components, only a few of them are not null. Under some assumptions on matrix G and the sparsity of $\mathbf{u}$, perfect recovery is possible even in situations where $M \ll N$. For our particular problem of similarity search, we want to exploit the following **key assumption**: Most images are *not* related to the query, whence most components of $\mathbf{u}$ are nearly 0 especially for high-dimensional spaces. On the contrary, the few images matching with the query produce positive similarities. Section 3.2 is devoted to this particular problem of similarity estimation with a sparsity assumption.

Notice that in very large scale database, storing the whole matrix G is not desirable, or not even possible. In practice, we simply store the group IDs each image belongs to, and the image IDs each group consists of. Section 5.1 gives a random construction of matrix G where each group has the same size $N_\mathcal{H}$, and each image belongs to the same number of groups $N_\mathcal{L}$. Specifically, when M is pre-fixed such that N divides $N_\mathcal{H}M$, $N_\mathcal{L}$ is guaranteed to be an integer such that

$$N_\mathcal{L} = N_\mathcal{H} \times \frac{M}{N}. \tag{6}$$

3. SCORE ESTIMATION STRATEGY

This section presents three score estimation strategies, namely based on pseudo-inverse, sparse decoding and group testing. They are introduced in a progressive manner, from the more standard ones to our group testing recovery method.

3.1 Pseudo-inverse PI

A standard method to solve (5) is to employ the pseudo-inverse of the grouping matrix G:

$$\tilde{\mathbf{u}} = (G^\top G)^{-1} G^\top \mathbf{v}. \tag{7}$$

This estimation provides the solution satisfying (5) minimizing the ℓ_2 norm $\|\tilde{\mathbf{u}}\|$ of the estimated similarity vector.

Albeit standard, this strategy has several drawbacks, both in terms of efficiency and accuracy. First, computing equation[2] (7) is time consuming: although the pseudo-inverse is fixed and computed offline, it is not sparse and multiplying it by a vector requires $N \times M$ elementary computations. Since M may be larger than D, especially for large databases, there is no computational advantage of using this strategy compared to the exhaustive search, whose complexity is $N \times D$. The lack of sparsity of the pseudo-inverse has another consequence: the variance of the similarity estimator for both the relevant and irrelevant images is large. This remark is related to the theoretical discussion provided in Section 4. Finally, this estimation does not exploit any particular assumption on $\mathbf{u}$, that is, the similarities associated with relevant images are expected to be significantly larger than the other measurements, *i.e.*, the distribution of the scores is closely distributed around 0, but has a heavy tail.

For these reasons, this method must be understood as a baseline, with respect to estimation quality, for the strategies that we propose in the rest of this section.

[2]We denote by $\mathbf{u}$ the true similarity vector, which, in our case, is the cosine similarity. We use $\tilde{\mathbf{u}}$ to denote the estimated image score vector.

3.2 Sparse decoding SD

As long as $\mathbf{u}$ is sparse enough, sparse decoding is capable of recovering $\mathbf{u}$ by solving the equation

$$\tilde{\mathbf{u}} = \min_{\mathbf{u}} \|\mathbf{v} - G\mathbf{u}\| + \lambda \|\mathbf{u}\|_1, \tag{8}$$

where $\|\mathbf{u}\|_1$ is the l_1-norm penalty added to regularize $\|\mathbf{v} - G\mathbf{u}\|$ [19]. The penalty ensures that most components of $\mathbf{u}$ are stuck around 0, hence providing a sparse solution.

In practice, irrelevant images have small similarity values and we do not really care about an exact recovery of these values. We are only interested in spotting large positive similarity values, while large negative similarities are assumed unlikely to occur. Therefore, we add positivity constraints on the solution $\mathbf{u}$,

$$\begin{aligned} \tilde{\mathbf{u}} = \min_{\mathbf{u}} \; & ||\mathbf{v} - G\mathbf{u}|| + \lambda \sum u_j. \\ s.t. \quad & u_j \geq 0 \end{aligned} \tag{9}$$

Nevertheless, we are facing the following difficulties: the sparsity of vector $\mathbf{u}$, defined as the number of large components and presumably also equal to the number of relevant database images, depends a lot on the query. Moreover, as the similarities of the irrelevant images are not strictly 0, we are in the context of noisy sparse signals recovery. Therefore, we can not perfectly reconstruct $\mathbf{u}$ but only aim at producing an estimation $\tilde{\mathbf{u}}$.

The design of matrix G also has its importance, because accurate recovery is guaranteed only if G satisfies the Restricted isometry property (RIP). Verifying this property for a given matrix G is NP-hard in general.

There are probabilistic constructions giving birth to RIP compliant matrices almost surely: for instance, generating its entries according to an i.i.d. sub-Gaussian distribution providing that $M = \mathcal{O}(W \log N)$, where W is an upper bound of the sparsity of $\mathbf{u}$. However, these sensing matrices are not sparse at all and difficult to handle for our large scale application. Note that algorithms solving (9) are iterative; each iteration requires a vector multiplication by matrix G.

To achieve a better trade-off between efficiency and effectiveness, we keep the matrix G sparse, while getting away from the compressed sensing framework: we have no guarantee on the reconstruction quality since our setup a priori does not necessarily fulfill the RIP and sparsity conditions. Yet, we find experimentally that, despite the inaccuracy of sparse decoding, nearly all the relevant images are assigned with large values in the estimation $\tilde{\mathbf{u}}$. Therefore, the sparse decoding algorithm is useful to indicate likely relevant images. We no longer see $\tilde{u}_j$ as the estimation of the similarity u_j but as a quantity indicating the likelihood that the j^{th} image matches the query, *i.e.* u_j is somehow 'big'.

Re-ranking. We calculate the true similarities between query and R images with largest likelihood scores $\tilde{u}_j$. By

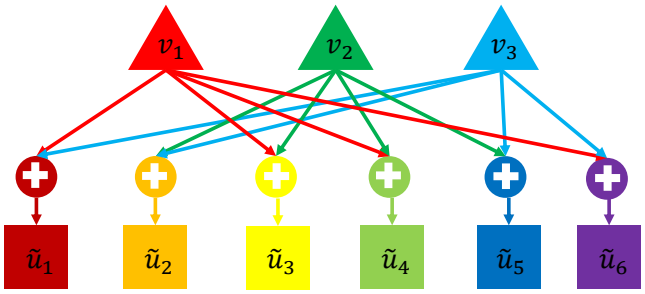


Figure 3: The basic score estimation strategy proposed in the group testing framework. v_1, v_2 and v_3 denote the corresponding group similarities in Figure 2. The image likelihood score $\tilde{u}_j$ is initially estimated by the sum the similarities of those groups that image j is connected with, *e.g.*, $\tilde{u}_1 = v_1 + v_3$.

revisiting only a fraction of image vectors ($R \ll N$), we are able to remove those irrelevant images and retain close to state-of-the-art retrieval accuracy. From this point view, the idea is closer to the adaptive group testing described in Figure 1 than to compressed sensing. The quantity $v_i = q^\top y_i$ is regarded as a test over the group of images $\mathcal{H}_i$ indicating whether some images in this group match the query.

3.3 Group testing GT

Revisiting the image vectors adds complexity and slows down the search. To relieve this concern, we further simplify the first stage discarding any iterative sparse decoding algorithm but employing the group testing approach. As illustrated in Figure 3, the 'likelihood' score for each image is estimated by summing the corresponding group similarities it belongs to:

$$\tilde{u}_j = \sum_{i \in \mathcal{L}_j} v_i, \tag{10}$$

where $\mathcal{L}_j$ denotes the set of groups image j belongs to. Vector-wise:

$$\tilde{\mathbf{u}} = G^\top \mathbf{v}. \tag{11}$$

Thanks to the sparsity of matrix G, the computational complexity is $M \times N_\mathcal{H}$, which is significantly smaller than $M \times N$.

Once again, $\tilde{\mathbf{u}}/N_\mathcal{L}$ is an estimation of $\mathbf{u}$ whose variance is relatively large. Despite its inaccuracy, the relevant images are usually assigned with large values in the estimation $\tilde{\mathbf{u}}$. To confirm the relevance of an image j with a large score $\tilde{u}_j$, we directly calculate its true similarity with the query vector.

This is the key: in order to improve the overall estimator, we consider an *adaptive* group testing strategy, which takes advantage of the direct measurement between the query and a few database images. This is done by an online back propagation update. More precisely, we take out the highest score from $\tilde{\mathbf{u}}$:

$$n = \arg\max_j \tilde{u}_j, \tag{12}$$

and in turn we calculate the exact similarity value

$$u_n = q^\top x_n \tag{13}$$

between query q and image n. Now that we have the true knowledge of n^{th} image's similarity to query, in order to eliminate its influence on those groups it is attached to, we subtract u_n from the initial group similarity $v_m^0 = v_m$ to yield an updated similarity v_m^1:

$$v_m^1 = v_m^0 - u_n = q^\top \left(\sum_{j \in \mathcal{H}_m^0} x_j - x_n \right) \tag{14}$$

$$= q^\top \sum_{j \in \mathcal{H}_m^0 \setminus \{n\}} x_j, \tag{15}$$

where m denotes the m^{th} group connected with image n. The new estimator v_m^1 is more accurate than v_m^0 to infer a possible match amongst the remaining images inside the group. We update the $N_\mathcal{L}$ group similarities connected with image n following (14). Subsequently, we back propagate the updated group similarities to their images as in Figure 3. The initial estimated image score vector $\tilde{\mathbf{u}}^0 = \tilde{\mathbf{u}}$ (11) is updated to $\tilde{\mathbf{u}}^1$. We repeat this update R times:

$$v_m^R = v_m^{R-1} - u_n = q^\top \sum_{j \in \mathcal{H}_m^{R-1} \setminus \{n\}} x_j. \tag{16}$$

At each iteration, the selected image of highest score is excluded as well as its connections with groups. Thereby, n is not the same in each update, the corresponding groups (m, $\mathcal{H}_m^{R-1}$) it connects with might also be different. The updated group similarities are back propagated to their images at each iteration. After $\tilde{\mathbf{u}}^{R-1}$ is updated to $\tilde{\mathbf{u}}^R$, we already calculate the cosine similarities of R images in total to the query, and they are ranked by their true similarity values during the back propagation stage.

Batch processing. In practice, when R is large, calculating and back-propagating the true similarity one vector by one affects the efficiency. To overcome this problem, we do not select only one image with the largest score, but instead the $\frac{R}{t}$ largest scores and propagate these messages at once, where t is a parameter setting the desired number of back propagations steps (instead of R). With t being small enough, the propagation phase is significantly accelerated and the bottleneck becomes the vector-level operations, *i.e.*, the query-group measurements and the direct measurements between the query and the selected images.

Computational cost. The overall computational cost in the proposed method is summarized as

$$\begin{aligned} T &= \mathcal{O}(M \times D) + \mathcal{O}(M \times N_\mathcal{H}) + \mathcal{O}(R \times D) \\ &\simeq \mathcal{O}((M + R) \times D) = \mathcal{O}(2M \times D). \end{aligned} \tag{17}$$

T can be approximated to $\mathcal{O}(2M \times D)$ as $D \gg N_\mathcal{H}$ and $R = M$. Compared to the baseline time complexity,

$$T_\text{B} = \mathcal{O}(N \times D), \tag{18}$$

the retrieval efficiency is guaranteed ($T \ll T_\text{B}$), *i.e.* as long as $2M \ll N$. We ignore the complexity of back propagation. $\frac{R}{t}$ images' true similarities are calculated each time, supposing all the M groups' similarities are correspondingly updated, then the additional operations will amount to $t \times M \times N_\mathcal{H}$ at most for t updates. As long as t is small, it is still negligible in high-dimensional spaces (large D) and large scale database (large N).

Variant (GT-V). In the Appendix, we present a simplified version of our GT-based estimator, denoted by GT-V (GT Variant). It works in the particular case that there

exists highly relevant database images to a query, such as near duplicate images. When we take out the largest score from $\tilde{\mathbf{u}}$ (11), we will find that $\tilde{u}_n$ is clearly larger than the other $\tilde{u}_j$. We could directly subtract $\tilde{u}_n/N_{\mathcal{L}}$ from the group similarities without calculating the true similarity, since we have a high confidence that the n^{th} image is highly relevant to the query. The corresponding image scores are updated following the same protocol with GT. In the end, we do t times back propagation with each time only one estimated image similarity being subtracted from the group similarities. We select the top-R ranked images with the largest scores from $\tilde{\mathbf{u}}^t$ and re-rank them according to their cosine similarities with query. The performance can reach the same level with GT, however, the additional computation during back propagation is further reduced to $t \times N_{\mathcal{H}} \times N_{\mathcal{L}}$.

Notice that GT-V is only effective for the databases that have highly relevant images with the query (perhaps the query's duplicate matched images). Notwithstanding GT-V is not a general method, it is almost as effective as GT for all the databases used in our experimental investigation.

4. DISCUSSION

We introduce a very simple probabilistic model to get some hints on how to tune our setup. By no means, we pretend that this model closely reflects reality. It is only a toy example illustrating a given phenomena, known as dilution in the group testing literature.

4.1 Model of the image similarity

Let x_j and q be two uniformly distributed random unit vectors of D dimension. We assume that two irrelevant images produce two statistically independent vectors. Denote by U_j the random variable modeling their cosine similarity u_j. It is asymptotically (for large D) distributed as a Gaussian law with null expectation and variance $1/D$: $U_j \sim \mathcal{N}\left(0, \frac{1}{D}\right)$. It indicates that, the larger D is, the sparser $\mathbf{u}$ will be, the easier it will be reconstructed.

When the images are relevant, we model the distribution of U_j by another Gaussian distribution: $U_j \sim \mathcal{N}(\mu, \sigma^2)$ with a positive expectation $\mu > 0$. We define $\mathcal{G}(x; \mu, \sigma^2) = \exp(-(x-\mu)^2/2\sigma^2)/\sqrt{2\pi}\sigma$ the probability density function (pdf) of $\mathcal{N}(\mu, \sigma^2)$.

4.2 Model of the group similarity

The distribution of the group similarity is a mixture of the above Gaussian distributions depending on how many relevant images there are in the group.

Suppose that there are c images matching with the query in the database of size N. Denote by K the random variable modeling the number of matching images in a given group of size $N_{\mathcal{H}}$. Since the group has been created at random, K follows a hypergeometric distribution:

$$\mathbb{P}[K=k] = \begin{cases} \frac{\binom{c}{k}\binom{N-c}{N_{\mathcal{H}}-k}}{\binom{N}{N_{\mathcal{H}}}} & \text{if } 0 \le k \le \min(c, N_{\mathcal{H}}), \\ 0 & \text{otherwise.} \end{cases} \quad (19)$$

We denote this probability by $P_N^c(k, N_{\mathcal{H}})$.

We assume now that the similarities of the images composing the group are mutually independent, which is reasonable since the group has been composed randomly. When there are k matches in the group, the random variable V related to the group similarity v is then distributed as $\mathcal{N}(k\mu, (N_{\mathcal{H}} - k)/D + k\sigma^2)$. In other words, its pdf is

$$f(v|K=k) = \mathcal{G}(x; k\mu, (N_{\mathcal{H}} - k)/D + k\sigma^2). \quad (20)$$

When we do not know how many matches there are in a group, the pdf of V is as follows:

$$\begin{aligned} f(v) &= \sum_{k=0}^{N_{\mathcal{H}}} \mathbb{P}[K=k] \cdot f(v|K=k) \\ &= \sum_{k=0}^{N_{\mathcal{H}}} P_N^c(k, N_{\mathcal{H}}).\mathcal{G}\left(x; k\mu, \frac{N_{\mathcal{H}}-k}{D} + k\sigma^2\right). \end{aligned}$$

4.3 Hypothesis test

The philosophy of group testing is to infer about the relevance of image j from the group similarities. We need to test the two following hypotheses:

Image j matches the query. When $g_{ij} = 1$, we know that image j belongs to group $\mathcal{H}_i$, therefore this group has at least one match. This knowledge transforms the pdf of the group similarity V_i into:

$$f_1^+(v) = \sum_{k=1}^{N_{\mathcal{H}}} P_{N-1}^{c-1}(k-1, N_{\mathcal{H}}-1).\mathcal{G}\left(x; k\mu, \frac{N_{\mathcal{H}}-k}{D} + k\sigma^2\right)$$

In the same way, when $g_{ij} = 0$, this knowledge transforms the pdf of the group similarity V_i into:

$$f_0^+(v) = \sum_{k=0}^{N_{\mathcal{H}}} P_{N-1}^{c-1}(k, N_{\mathcal{H}}).\mathcal{G}\left(x; k\mu, \frac{N_{\mathcal{H}}-k}{D} + k\sigma^2\right).$$

Image j does not match the query. When $g_{ij} = 1$, we know that this group cannot be composed of matching images only:

$$f_1^-(v) = \sum_{k=0}^{N_{\mathcal{H}}-1} P_{N-1}^{c}(k, N_{\mathcal{H}}-1).\mathcal{G}\left(x; k\mu, \frac{N_{\mathcal{H}}-k}{D} + k\sigma^2\right).$$

When $g_{ij} = 0$, this knowledge transforms the pdf of the group similarity V_i into:

$$f_0^-(v) = \sum_{k=0}^{N_{\mathcal{H}}} P_{N-1}^{c}(k, N_{\mathcal{H}}).\mathcal{G}\left(x; k\mu, \frac{N_{\mathcal{H}}-k}{D} + k\sigma^2\right).$$

Which groups bring information. In reality, we do not know wether image j is a match. If it is, $N_{\mathcal{L}}$ group similarities are distributed according to f_1^+ and $M - N_{\mathcal{L}}$ according to f_0^+. If it is not, then $N_{\mathcal{L}}$ group similarities are distributed according to f_1^- and $M - N_{\mathcal{L}}$ according to f_0^-. Since $c \ll N$, we have indeed that

$$f_0^+(v) \approx f_0^-(v), \quad \forall v \in \mathbb{R}. \quad (21)$$

This means that knowing whether image j matches the query or not has no impact on the similarities of the groups which image j doesn't belong to. It is only the similarities of the $N_{\mathcal{L}}$ groups where image j is, that bring some information about the match with the query.

4.4 Amount of information

The group similarities provide an amount of information about the match of image j which can be measured by the Kullback Leibler distance between their distributions under

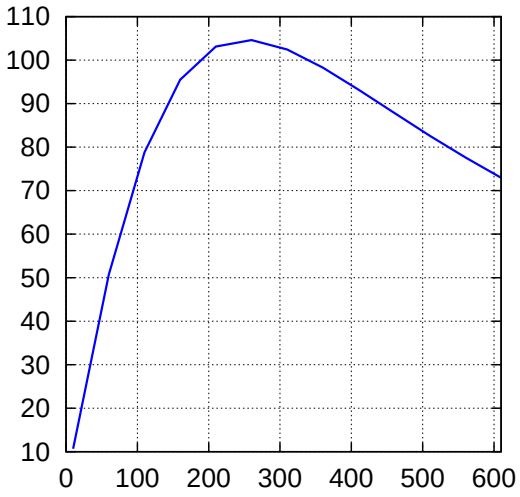


Figure 4: Total Kullback Leibler distance in nats, as a function of $N_{\mathcal{H}}$. Setup: $D = 1920$, $\mu = 0.47$, $\sigma^2 = 0.16$, $c = 3$, $n = 10^6$.

both hypothesis. We assume that the group similarities are statistically independent (which is not exactly true). The total Kullback Leibler distance between the distribution of the M group similarities under both hypothesis is simply (thanks to approximation (21)):

$$D_{KL} = N_{\mathcal{L}} \cdot D(f_1^+ || f_1^-) = M \frac{N_{\mathcal{H}}}{N} \cdot D(f_1^+ || f_1^-), \qquad (22)$$

where $D(f_1^+ || f_1^-)$ is the Kullback-Leibler distance between pdf f_1^+ and f_1^-.

This last quantity heavily depends on $N_{\mathcal{H}}$: The bigger $N_{\mathcal{H}}$, the bigger the variances of the involved Gaussian distributions (20), the smoother and more similar f_1^+ and f_1^-, and the lower $D(f_1^+ || f_1^-)$. This means that the quantity of information brought by one group similarity about the matching of a given image in the database is getting lower. This phenomenon is known in group testing as the dilution: composing groups with too many items dilute the detectability of the match of one particular item.

However, this phenomenon is partly compensated by the fact that, for a fixed ratio M/N, larger groups (*i.e.* big $N_{\mathcal{H}}$) implies that a given image is present in more groups (*i.e.* big $N_{\mathcal{L}}$ thanks to (6)). Each group brings less information, but we have more groups to detect whether a particular image is a match. In the end, it is not possible to theoretically derive the best value of $N_{\mathcal{H}}$ maximizing the total Kullback Leibler distance D_{KL}. Figure 4 plots this quantity for this toy example showing that there is an optimal value for $N_{\mathcal{H}}$. In the experimental session, we will show the optimal value of $N_{\mathcal{H}}$ in the enumerable test of $N_{\mathcal{L}}$ as appearing in (6).

5. EXPERIMENTS

This section evaluates and compares the algorithms proposed in our framework. We consider an image retrieval scenario. We, first, present the evaluation protocol and give some implementation details. Then we discuss the parameters and analyze the results.

5.1 Dataset and evaluation protocol

We use public benchmarks commonly used by papers on image retrieval for the evaluation.

INRIA Holidays is a set of images which mainly contains holidays photos [9]. It includes a large variety of scene types (natural, man-made, water and fire effects, etc.) and has 500 queries and 1,491 images in total. The evaluation is carried out in a leave-one-out manner, meaning the query image is retrieved from the set prior to retrieval.

UKB. The University of Kentucky Benchmark dataset [21] consists of 10,200 images grouped into 2,550 subsets of corresponding images. Each subset contains four images. For a given query, the system is expected to return the four relevant images in the first four positions.

Flickr1M. The Flickr1M dataset consists of 1 million images downloaded from Flickr [23]. The images have been downloaded by searching Flickr and are used as distractors in our experiment.

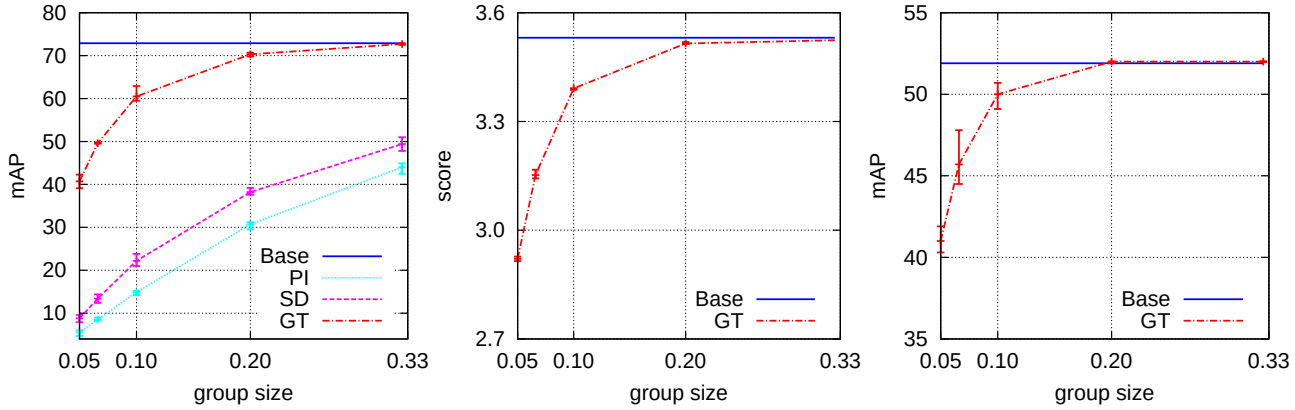
Evaluations are first conducted on a small and medium-sized datasets (Holidays and UKB). To evaluate the performance on larger scale, we gradually add images from the Flickr1M dataset as distractors. In this case, the evaluation is performed on the 500 Holidays queries, since the images from Flickr1M are not relevant to the queries. Note that some methods, like pseudo-inverse and sparse decoding, are not evaluated on the larger sets because they are not tractable: although they use less group-to-query operations, deducing the image similarities is their bottleneck.

Evaluation protocol. The performance for the Holidays and Flickr1M datasets is measured in terms of the average precision (AP), which is defined as the area under the precision-recall curve [23] for each query. The AP score is computed for each query and averaged to obtain a mean average precision (mAP). For the UKB dataset, the score is standardly computed as the average number of correct images in the top-4 positions (4-recall@4), the best score is 4. In the same test, there is always a small variance of the performance due to the randomness of the group matrix. We report here the median value (maximum and minimal values for the variances) out of 5 repetitions. The baseline is obtained from a recent paper [13].

5.2 Implementation notes

Design of Matrix G. Referring to the model discussion session, $N_{\mathcal{H}}$ and $N_{\mathcal{L}}$ have a strong impact on the variances of the group similarities. These parameters should not be very large, so that the detectability of the match of one particular relevant image in a group is not diluted too much, as discussed in Section 4.

The parameter $N_{\mathcal{H}}$ is chosen to be a small value, Figure 6 shows a simple example on how to construct the matrix: starting from a $M \times N$ zero matrix $\widetilde{G}$, we set some values to 1s: $N_{\mathcal{H}}$ elements (red) in the first row as one, and $N_{\mathcal{H}}+1 : 2N_{\mathcal{H}}$ elements (green) in the second row as one, *etc.* When it comes to the end, we restart from the beginning (brown) until M rows are initialized. This construction guarantees that each group has approximately the same number of images $N_{\mathcal{H}}$, and each image belongs to approximately the same number of groups $N_{\mathcal{L}}$. To make it



(a) Holidays (b) UKB (c) Holidays+Flickr1M

Figure 5: Group size varied in the range of $\{\frac{1}{20}, \frac{1}{15}, \frac{1}{10}, \frac{1}{5}, \frac{1}{3}\}$ of the database size. $N_{\mathcal{L}}$ is chosen as 2 according to Table 1. Vocabulary size is 16.

Table 1: Performance as a function of the number of groups per image $N_{\mathcal{L}}$ on Holidays (+ Flickr1M) and UKB datasets. $N_{\mathcal{L}}$ varies from 1 to 20, while M is 1/10 of database size. The results are given for *(top)* 8064- and *(bottom)* 1920-dimensional image descriptors.

$N_{\mathcal{L}}(k_c = 64)$	Holidays	UKB
	GT	GT
1	68.3	3.56
2	**69.6**	**3.57**
5	68.3	3.56
10	63.8	3.56
20	60.3	3.56
baseline	77.2	3.63

$N_{\mathcal{L}}(k_c = 16)$	UKB	Holidays+Flickr1M
	GT	GT
1	3.37	49.4
2	**3.39**	**50.0**
5	3.37	49.2
10	3.36	49.2
20	3.37	48.3
baseline	3.53	51.9

random, we apply some random permutations on the matrix columns, and consequently, we could produce a random matrix satisfying (6). Note that this construction is similar to the one considered in a paper for projecting sparse vectors [10].

Experimental setup. The local descriptors are extracted with the Hessian-affine detector and described by SIFT [15]. We use the RootSIFT variant [1, 8] in all our experiments. Overall, the descriptor extraction follows the procedure in our baseline [13]. The triangulation embedding and democratic aggregation [13] are used to aggregate the features with a vocabulary size k_c being set to either 16 or 64. We obtain the vocabulary independently from the dataset Flickr60k. Given the SIFT descriptor's length of 128, the dimension

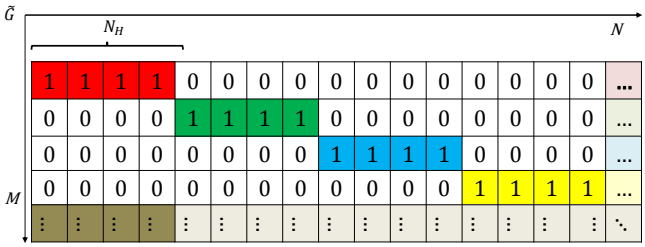


Figure 6: An illustration of how to construct the group matrix.

of our representation is thus $128 \times 16(64) = 2,048(8,192)$. In real implementation, the first 128 components associated with the largest eigenvalues are discarded to reduce the interferences between the coded descriptors. The final dimensionality D of the representation is therefore $1,920(8,064)$. The number of groups M is by default $\frac{1}{10}$ of the database size, and we calculate the cosine similarities of the same number of images in back propagation stage ($R = M$). Considering both the efficiency and effectiveness, t is set to 10 in GT. $N_{\mathcal{L}}$ varies in different datasets and score estimation strategies, as discussed in the following subsection evaluating the impact of the parameter. The quantity $N_{\mathcal{H}}$ is obtained by (6).

5.3 Impact of the parameters

Number of groups per image $N_{\mathcal{L}}$. Table 1 shows the performance of varying number of groups per image in Holidays, UKB and Flickr1M datasets. The best $N_{\mathcal{L}}$ is 2 for both vocabulary size and dataset. The performance is not very sensitive to $N_{\mathcal{L}}$, we can generally take it as small as 2 for better efficiency.

Total number of groups M. For a fixed value of $N_{\mathcal{L}}$, we evaluate the impact of the total number of groups in Holidays, UKB, and Holidays + Flickr1M datasets. We vary M from 1/20 to 1/3 of the database size, and obtain similarity scores reported in Figure 5 and Table 2 for vocabulary sizes $k_c = 16$ and 64. Increasing the group size appear to help. Indeed, with M simply being 1/10 of database size and

Table 2: The number of groups M varies in the range of $\{\frac{1}{20}, \frac{1}{15}, \frac{1}{10}, \frac{1}{5}, \frac{1}{3}\}$ of the database size N. $N_{\mathcal{L}}$ is chosen as 2 according to Table 1. The vocabulary size is $k_c = 64$.

$\frac{M}{N}$	Holidays	UKB
	GT	GT
1/20	52.5	3.32
1/15	59.7	3.46
1/10	69.5	3.57
1/5	75.5	3.63
1/3	77.2	3.63
baseline	77.2	3.63

$k_c = 16$, the mAP using back propagation reaches around 96% of the baseline benchmark in Holidays+Flickr1M. This means that we only need 2/10 (another 1/10 is for the back propagation) of the original computational cost to recover 96% of the baseline precision.

Pseudo-inverse and sparse decoding results on Holidays dataset are also presented. They are more time consuming: it is not feasible to carry out them for a large dataset. For pseudo-inverse, it only works when the matrix G is very dense ($N_{\mathcal{L}}$ is very large), which makes the reconstruction very inefficient. For sparse decoding, we add positive constrains in the coefficients of $\mathbf{u}$ (9), otherwise, the mAP is a bit higher.

As previously reported [13], the performance improves by increasing the vocabulary size k_c from 16 to 64. In our framework, it is important to use a large dimensionality: the dimensionality D ($k_c \times 128 - 128$) significantly impacts the level of noisy variance in group testing. Increasing D has a beneficial on this variance (see Section 4.1), and the overall performance is therefore improved.

Back propagation times. Table 3 shows the results, for UKB and Holidays+Flickr1M datasets, with different numbers of back-propagation stages t . GT is improved with a larger number of back propagation iterations. Setting $t = 10$ appears a good compromise both in terms of efficiency and effectiveness. Take Holidays+Flickr1M as an example, we apply $t = 10$ back propagations steps, each involving direct measurements for $10k$ images (those with the highest ranks). Assuming that all group similarities are updated by one back propagation, the additional computation is thus approximated to $t \times (10k \times N_{\mathcal{H}}) = 20 \times N$, where $N_{\mathcal{H}}$ is 20 when $N_{\mathcal{L}}$ is chosen as 2 for the best. Compared to the $N \times D$ computations of the baseline, the overall increase in efficiency is still guaranteed because $D = 1,920 \gg 20$.

We also present the result for GT-V. The performance drop for large values of t. In our opinion, this is because, for UKB and Holidays+Flickr1M, there is only one duplicate image per query (the query itself). Refer to the Appendix for more details about this variant.

Group *vs* back propagation. Assuming that $M + R = 100k$ is fixed in Holidays + Flickr1M, Figure 7 analyzes the trade-off between M and R. We achieve the best performance for $M = R = 50k$. The default setting follows this observation. It is also interesting that, 1) the performance is obviously better with a larger ratio M/R; 2) when $M = 100k$ and $R = 0$ (no back propagation), the mAP is only 3.3.

Table 3: Performance as a function of the number of back propagation stages t (Holidays + Flickr1M and UKB datasets). The number of groups is set by default to $\frac{N}{10}$ and $N_{\mathcal{L}}$ is the optimal value. $k_c = 16$.

t	UKB		Holidays+Flick1M	
	GT	GT-V	GT	GT-V
0	3.28		48.9	
1	3.34	**3.34**	49.5	**49.4**
4	3.38	2.04	48.9	19.6
8	3.37	2.05	49.5	16.3
10	**3.39**	2.04	**50.0**	16.4

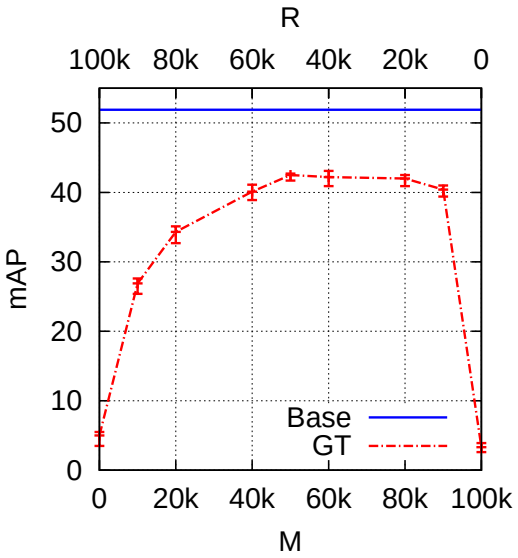


Figure 7: Performance (mAP) on Holidays + Flickr1M dataset corresponding to different combinations of M and R when $M + R$ is fixed to $100k$ and vocabulary size $k_c = 16$.

5.4 Overall performance

Database size. Figure 8 reports the mAP values obtained when we gradually add images from Flickr1M as distractors to Holidays dataset. For larger databases, the mAP of our methods approaches the baseline. This behavior is expected because group testing is more effective when the number of positive tests is small compared to the total number of tests: there is comparatively less interferences in the score estimation. This advocates the practical interest of our group testing framework for very large scale image retrieval.

The results of GT-V are also given in this figure as an alternative choice. Although GT-V is not provably good in general, it is competitive in all the databases we use, for there are always images that are highly relevant to the query. Nevertheless, we generally observe some loss compared to our main GT approach.

PCA dimensionality reduction. Another way to reduce the computational cost is to use dimensionality reduction. We demonstrate that our approach is complementary with PCA dimensionality reduction. To evaluate this, we keep the first D' components of the representation (as evaluated

Table 4: Performance after dimensionality reduction to short vectors. This evaluation follows our default settings. There are two set of descriptors used in rows 3–6 and 7–9, corresponding to vectors of dimensionality 1920 and 8064, respectively. We compare the methods PCA, GT, and P+G (PCA + GT) for two different operating points, aiming at reducing the complexity of the baseline by a factor 8 (columns 3 to 5) and 32 (columns 6 to 8). The number of operations is reflected by $N_s \times D'$, where N_s denotes the number of vector comparisons. The combination of PCA with group testing always achieves the best result for a fixed complexity.

Method	Base	PCA	GT	P+G	PCA	GT	P+G
N_s	N	N	$\frac{N}{8}$	$\frac{N}{2}$	N	$\frac{N}{32}$	$\frac{N}{2}$
D'	1920	$\frac{1920}{8}$	1920	$\frac{1920}{4}$	$\frac{1920}{32}$	1920	$\frac{1920}{16}$
Holidays	72.8	65.6	49.1	**67.2**	56.2	7.3	**60.3**
UKB	3.53	3.45	3.08	**3.48**	3.26	1.55	**3.33**
Flickr1M	51.9	42.9	44.6	**46.6**	32.0	15.9	**39.8**
D'	8064	$\frac{8064}{8}$	8064	$\frac{8064}{4}$	$\frac{8064}{32}$	8064	$\frac{8064}{16}$
Holidays	77.2	71.7	60.7	**73.8**	65.6	20.3	**68.5**
UKB	3.63	3.59	3.41	**3.62**	3.52	2.31	**3.56**

in our image retrieval baseline [13]): Table 4 reports the performance with different D', k_c and N_s. Our goal is to measure the performance for a fixed overall complexity. N_s denotes the total number of vector comparisons, *e.g.*, in GT, $N_s = R + M$, it is $\frac{N}{8}$ meaning that $\frac{N}{16}$ for the M group similarities and another $\frac{N}{16}$ for the R image similarities in the back propagation. The overall computations in different methods is approximately summarized by the cost $N_s \times D'$.

Compared to the baseline, there is a drop in performance in GT due to the smaller number of measurements (and therefore small number of computations). The same remark holds for the PCA: if we divide the dimensionality by 8 and 32 in PCA, so that the computation of PCA is the same with that of GT, the performance decreases. Yet, our method is better than PCA on the large Holidays+Flickr1M dataset. For the smaller datasets, *i.e.* UKB and Holidays, it is inferior but could be close to PCA as long as N_s is not too small and k_c is large. In essence, these experiments show that the proposed group testing framework are comparatively more effective for large scale database and high-dimensional spaces.

We also conduct the experiment by combining PCA with GT (P + G). First, we divide the original dimensionality by 4, and 16; second, we embed GT by setting M as $\frac{N}{4}$ ($R = M$), which means that N_s is $\frac{N}{2}$. Considering the results that achieves the same overall complexity, it appears that the combination of PCA with GT outperforms both PCA and GT. Again, the performance is comparatively better on the large scale databases and high-dimensional spaces.

6. CONCLUSION

Our framework for similarity search in high-dimensional spaces amounts to replacing the individual query-database similarity computations by group measurements, in a way strongly inspired by adaptive group testing algorithms. In this context, the key algorithmic component is to estimate the individual image similarities from the group measurements. The first conclusion of our paper is that sparse recovery algorithms with positivity constraints, such as those used in compressed sensing, already give better results than the regular pseudo-inverse method. Yet this strategy has a limited practical interest due to its high computational cost.

Moreover, sparse recovery makes the (invalid) assumption that most similarities are zeros. For this reason, our main strategy based on adaptive group testing demonstrates a better performance, especially for large datasets and for the higher-dimensional spaces. As a result, our strategy offers competitive performance with respect to the compromise between retrieval efficiency and accuracy, and is shown complementary to PCA dimensionality reduction.

As a final note, we mention that this exploratory strategy is certainly not optimum with respect to the achievable trade-offs. From a group testing perspective, our algorithm is very simple and does not formally optimize the accuracy under a constraint on the number of measurements. Similarly, the group design is not directly optimized. For these reasons, we believe that further possible improvement is possible in this promising framework.

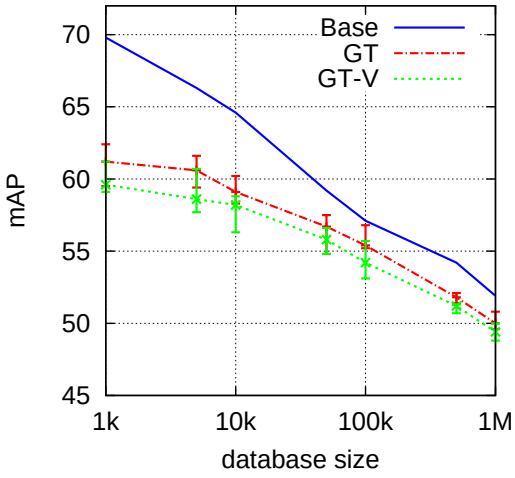


Figure 8: mAP values corresponding to different sizes of Holidays + Flickr (1k, 5k, 10k, 50k, 100k, 500k, 1M). Group size is $\frac{1}{10}$ of the database size while vocabulary size is 16.

7. ACKNOWLEDGEMENTS

This work was supported by ERC grant VIAMASS no. 336054 and ANR project FIRE-ID. Miaojing Shi was supported by NBRPC 2011CB302400, NSFC 61121002, 61375026 and JCYJ 20120614152136201, he thanks his supervisor Prof. Chao Xu for encouraging him to work on this paper.

8. REFERENCES

[1] R. Arandjelovic and A. Zisserman. Three things everyone should know to improve object retrieval. In *CVPR*, June 2012.

[2] R. Balu, T. Furon, and H. Jégou. Beyond "project and sign" for cosine estimation with binary codes. In *ICASSP*, 2014.

[3] M. S. Charikar. Similarity estimation techniques from rounding algorithms. In *STOC*, pages 380–388, May 2002.

[4] M. Datar, N. Immorlica, P. Indyk, and V. Mirrokni. Locality-sensitive hashing scheme based on p-stable distributions. In *Proceedings of the Symposium on Computational Geometry*, pages 253–262, 2004.

[5] R. Dorfman. The detection of defective members of large populations. *The Annals of Mathematical Statistics*, 14(4):436–440, December 1943.

[6] D.-Z. Du and F. K. Hwang. *Combinatorial group testing and its applications*. World Scientific, 1993.

[7] P. Indyk and R. Motwani. Approximate nearest neighbors: towards removing the curse of dimensionality. In *STOC*, pages 604–613, 1998.

[8] M. Jain, R. Benmokhtar, P. Gros, and H. Jégou. Hamming embedding similarity-based image classification. In *ICMR*, June 2012.

[9] H. Jégou, M. Douze, and C. Schmid. Hamming embedding and weak geometric consistency for large scale image search. In *ECCV*, October 2008.

[10] H. Jégou, M. Douze, and C. Schmid. Packing bag-of-features. In *ICCV*, September 2009.

[11] H. Jégou, M. Douze, and C. Schmid. Product quantization for nearest neighbor search. *IEEE Trans. PAMI*, 33(1):117–128, January 2011.

[12] H. Jégou, F. Perronnin, M. Douze, J. Sánchez, P. Pérez, and C. Schmid. Aggregating local descriptors into compact codes. In *IEEE Trans. PAMI*, September 2012.

[13] H. Jégou and A. Zisserman. Triangulation embedding and democratic kernels for image search. In *CVPR*, June 2014.

[14] F. Krzakala, M. Mézard, F. Sausset, Y. Sun, and L. Zdeborová. Statistical-physics-based reconstruction in compressed sensing. *Physical Review X*, 2(2), 2012.

[15] D. Lowe. Distinctive image features from scale-invariant keypoints. *IJCV*, 60(2):91–110, 2004.

[16] Q. Luo, S. Zhang, T. Huang, W. Gao, and Q. Tian. Superimage: Packing semantic-relevant images for indexing and retrieval. In *ICMR*, April 2014.

[17] Q. Lv, M. Charikar, and K. Li. Image similarity search with compact data structures. In *CIKM*, pages 208–217, November 2004.

[18] Q. Lv, W. Josephson, Z. Wang, M. Charikar, and K. Li. Multi-probe LSH: Efficient indexing for high-dimensional similarity search. In *Proceedings of the International Conference on Very Large DataBases*, pages 950–961, 2007.

[19] J. Mairal, F. Bach, J. Ponce, and G. Sapiro. Online dictionary learning for sparse coding. In *ICML*, 2009.

[20] M. Muja and D. G. Lowe. Fast approximate nearest neighbors with automatic algorithm configuration. In *VISAPP*, February 2009.

[21] D. Nistér and H. Stewénius. Scalable recognition with a vocabulary tree. In *CVPR*, pages 2161–2168, June 2006.

[22] F. Perronnin and C. R. Dance. Fisher kernels on visual vocabularies for image categorization. In *CVPR*, June 2007.

[23] J. Philbin, O. Chum, M. Isard, J. Sivic, and A. Zisserman. Object retrieval with large vocabularies and fast spatial matching. In *CVPR*, June 2007.

[24] J. Sivic and A. Zisserman. Video Google: A text retrieval approach to object matching in videos. In *ICCV*, pages 1470–1477, October 2003.

[25] A. Torralba, R. Fergus, and Y. Weiss. Small codes and large databases for recognition. In *CVPR*, June 2008.

APPENDIX: Group testing variant GT-V

This appendix details the variant mentioned at the end of Section 3. It specifically assumes that some searched images (for instance duplicate images) in the database are very similar to the query. The groups connected with these highly relevant images have thus strong similarities, but the presence of other relevant images in these groups might be consequently hidden. To cancel this shadowing in these groups, we propose an alternative method: group testing variance.

Likewise (11) and (12), we obtain the initial scores $\tilde{\mathbf{u}}$ and spot the image with the highest score. Empirically, we observe that $\tilde{u}_n$ is significantly larger than the other $\tilde{u}_j$. This gives us a high confidence that the n^{th} image is highly relevant to the query, and all the groups in which it occurs have relatively larger similarities.

This n^{th} image may hide the presence of other relevant images in the groups of $\mathcal{L}_n$. We subtract $\tilde{u}_n^0 = \tilde{u}_n$ from initial group similarities $v_m^0 = v_m$, $m \in \mathcal{L}_n$, to yield an update similarity v_m^1:

$$v_m^1 = v_m^0 - \tilde{u}_n^0 . \frac{1}{|\mathcal{L}_n|} = q^\top \left(\sum_{j \in \mathcal{H}_m^0} x_j - \frac{1}{|\mathcal{L}_n|} \sum_{i \in \mathcal{L}_n} v_i^0 \right). \quad (23)$$

Update v_m^1 more precisely depicts the likelihood scores of the group composed of the remainging images.

Following the same protocol as in the main GT method, all the $N_{\mathcal{L}}$ group similarities connected with image n are updated and back propagated to the images (the n^{th} image being excluded) connected with these groups. The image scores in $\tilde{\mathbf{u}}$ is updated, and we repeat the update t times:

$$v_m^t = v_m^{t-1} - \tilde{u}_n^{t-1} . \frac{1}{|\mathcal{L}_n|}. \quad (24)$$

Afterwards, we rank the image list according to their values in $\tilde{\mathbf{u}}^t$ and re-rank the top-R ones by calculating their true cosine similarities.

The computational cost is comparable to GT, as discussed below for the two main steps of these algorithms.

1. largest score removal: once $N_{\mathcal{L}}$ connected group similarities are changed, the scores of the $N_{\mathcal{H}} \times N_{\mathcal{L}}$ images connected with these groups are updated. The additional computation thus amounts to $t \times N_{\mathcal{H}} \times N_{\mathcal{L}}$ operations for a number of t updates;
2. image list re-ranking: the computation is $\mathcal{O}(N \log R)$ for selecting the top-R ranked images, plus $\mathcal{O}(R \log R)$ for re-ranking them by their true similarities. With R equalling M and small enough, it is comparable to the time complexity of ranking the entire list $\mathcal{O}(N \log N)$.

If there is no highly relevant images in the database, the image with the largest estimated score might not truly matching the query. In this case, we suggest adopting the more reliable GT estimation.