"Clustering by Composition" – Unsupervised Discovery of Image Categories

Alon Faktor* and Michal Irani

Dept. of Computer Science and Applied Math
The Weizmann Institute of Science, ISRAEL

Abstract. We define a "good image cluster" as one in which images can be easily composed (like a puzzle) using pieces from each other, while are difficult to compose from images outside the cluster. The larger and more statistically significant the pieces are, the stronger the affinity between the images. This gives rise to unsupervised discovery of very challenging image categories. We further show how multiple images can be composed from each other simultaneously and efficiently using a collaborative randomized search algorithm. This collaborative process exploits the "wisdom of crowds of images", to obtain a sparse yet meaningful set of image affinities, and in time which is almost linear in the size of the image collection. "Clustering-by-Composition" can be applied to very few images (where a 'cluster model' cannot be 'learned'), as well as on benchmark evaluation datasets, and yields state-of-the-art results.

1 Introduction

A fundamental problem in computer vision is that of unsupervised discovery of visual categories in a collection of images. This is usually approached by applying image clustering to the collection, thereby grouping the images into meaningful clusters of shared visual properties (e.g. shared objects or scene properties). The clustering process aims to find the underlying structures and exploit them to partition the collection into clusters of "similar" images.

One of the first works on unsupervised category discovery [1] relied on pairwise affinities between images (using the Pyramid Match Kernel). However, these affinities are typically not strong enough to capture complex visual similarity between images. Consequently, more sophisticated methods have been proposed, which usually start with simple pairwise affinities, but refine these iteratively until a 'common cluster model' emerges. These can be common segments [2], common contours [3,4], common distribution of descriptors [5], representative cluster descriptors [6,7], etc.

However, observing the challenging image collection of Ballet and Yoga images in Fig. 2, there seems to be no single (nor even few) 'common model(s)' shared by all images of the same category. The poses within each category vary significantly from one image to another, there is a lot of foreground clutter (different clothes, multiple people, occlusions, etc.), as well as distracting backgrounds. Therefore, the above methods for *unsupervised* 'learning' of a shared 'cluster model' will most likely fail (not only due

* Funded in part by the Israeli Science Foundation and the Israeli Ministry of Science.

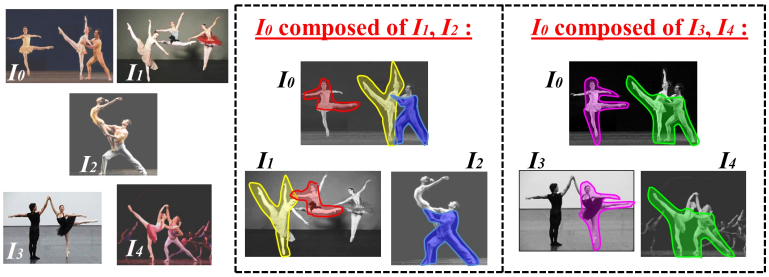

Fig. 1. Compositions used for computing image affinities.

to the large variability within each category, but also due to the small number of images per category). In the absence of an emerging 'common cluster model', these methods will be dominated by their simple initial pairwise affinities.

In this paper we suggest an approach, which does *not* seek a 'common cluster model', but rather uses *sophisticated images affinities* based on "Similarity by Composition" [8]. Although the ballet poses differ from each other, one ballet pose can be easily composed from pieces of other ballet poses (Fig 1). Our approach detects "statistically significant" regions which co-occur between images. "Statistically significant" regions are regions which have a low chance of occurring at random. The reoccurrence of such regions across images induces strong and meaningful affinities, even if they do not appear in many images (and thus can not be identified as a 'common model').

We define a "good image cluster" as one in which each image can be easily composed using statistically significant pieces from other images in the cluster, while is difficult to compose from images outside the cluster. We refer to this as "Clustering by Composition". We further show how multiple images can be composed from each other simultaneously and efficiently using a *collaborative randomized search algorithm*. Each image 'suggests' to other images where to search for similar regions within the image collection. This collaborative process exploits the "wisdom of *crowds of images*", to obtain a sparse yet meaningful set of image affinities, and in time which is almost linear in the size of the image collection. "Clustering by Composition" can be applied to very few images, as well as on benchmark datasets, and yields state-of-the-art results.

The rest of this paper is organized as follows: In Sec. 2 we provide a high-level overview of our approach, which is then detailed in Sections 3, 4, and 5. Experimental results can be found in Sec. 6.

2 Overview of the Approach

Image affinities by composition: Our image affinities are based on "Similarity by Composition" [8]. The notion of composition is illustrated in Fig 1. The Ballet image I_0 is composed of a few large (irregularly shaped) regions from the Ballet images I_1 and I_2. This induces strong affinities between I_0 and I_1, I_2. The larger and more *statistically significant* those regions are (i.e., have low chance of occurring at random), the stronger the affinities. The Ballet image I_0 could probably be composed of Yoga

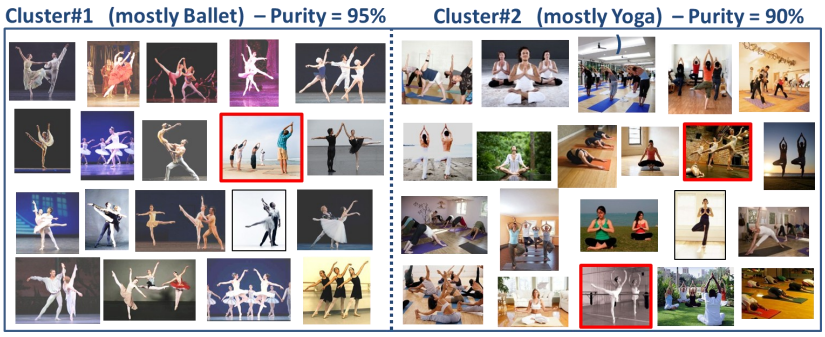

Fig. 2. Clustering Results on our Ballet-Yoga dataset. *This dataset contains 20 Ballet and 20 Yoga images (all shown here). Images assigned to the wrong cluster are marked in red. We obtain mean purity of* 92.5% *(*37 *out of* 40 *images are correctly clustered).*

images as well. However, while the composition of I_0 from other Ballet images is very simple (a 'toddler puzzle' with few large pieces), the composition of I_0 from Yoga images is more complicated (a complex 'adult puzzle' with many tiny pieces), resulting in low affinities. These affinities are quantified in Sec. 3 in terms of the "number of bits saved" by describing an image using the composition, as opposed to generating it 'from scratch' at random. To obtain reliable clustering, each image should have 'good compositions' from multiple images in its cluster, resulting in high affinity to many images in the cluster. Fig 1 illustrates two different 'good compositions' of I_0.

Note that "good regions" employed in the composition are typically NOT 'good image segments': they are not confined by image edges, may be a part of a segment, or contain multiple segments. Therefore, such regions cannot be extracted ahead of time via image segmentation (as in [2,9]). What makes them 'good regions' is NOT them being 'good segments', but rather the fact that they *co-occur* across images, yet, are *statistically significant* (non-trivial).

'Good regions' are *image-specific*, and not cluster-specific. A region may co-occur *only once*, yet still provide strong evidence to the affinity between two images. Such an infrequent region cannot be 'discovered' as a 'common cluster shape' from the collection (as in [3,4]). **Employing the co-occurrence of non-trivial large regions, allows to take advantage of *high-order statistics* and geometry**, even if infrequent, and without the necessity to 'model' it. Our approach can therefore handle also very small datasets with very large diversity in appearance (as in Figs. 2, 6).

Efficient "collaborative" multi-image composition: In principle, finding all matching regions of *arbitrary size and shape* is a very hard problem (already between a pair of images, let alone in a large image collection). We propose an efficient *collaborative randomized* algorithm, which *simultaneously* composes all images in the collection from each other. Images collaborate by guiding each other where to search for good regions. This algorithm detects with *very high probability* the statistically significant compositions in the collection, in runtime almost linear in the size of the collection.

Our randomized algorithm is inspired by "PatchMatch" [10], but searches for 'similar *regions*' (as opposed to similar patches or descriptors). We show that when randomly sampling descriptors across a pair of images, and allowing collaboration between descriptors, *large* shared regions (of unknown shape, size, or position) can be detected in linear time $O(N)$ (where N is the size of the image). In fact, *the larger the region, the faster it will be found, and with higher probability*. This leverages on the "wisdom of *crowds of pixels*". These ideas are formulated in Sec. 4.

Finding the statistically significant compositions inside a *collection* of M images should require in principle to go over all pairs of images - i.e. a complexity of $O(NM^2)$. However, we show that when *all* the images in the collection are composed *simultaneously* from each other, we can exploit the "wisdom of *crowds of images*" to iteratively generate the most significant compositions for each image (without having to go over all the image pairs). **Images 'give advice' to each other where to search in the collection** according to their current matches. For example, looking at Fig. 1, image I_0 has strong affinity to images $I_1, .., I_4$. Therefore, in the next iteration, I_0 can 'encourage' $I_1, .., I_4$ to search for matching regions in each other. Thus, e.g., I_3 will be 'encouraged' to sample more in I_1 in the next iteration. Note that the shared regions between I_1 and I_3 need not be the same as those they share with I_0. For example, the entire upper body of the standing man in I_3 is similar to that of the jumping lady in the center of I_1.

This process produces within a few iterations a *sparse* set of reliable affinities (corresponding to the most significant compositions). Such sparsity is essential for good image clustering, and is obtained here via 'collective decisions' made by all the images. The collaboration reduces the computational complexity of the overall composition dramatically, to $O(NM)$. In other words, the *average complexity per image* remains very small - practically linear in the size of the image $O(N)$, *regardless of the number of images M in the collection*! Sec. 4 explains the randomized region search and provides analysis of its complexity, while Sec. 5 describes the overall clustering algorithm.

3 Computing Image Affinities by Composition

'Similarity by Composition' [8] defines a similarity measure between a 'Query image' Q and a 'Reference image' Ref, according to the 'ease' of composing Q from pieces of Ref. Below we explain how we employ those ideas (modified and adapted to the needs of our algorithm) for computing affinities between images.

Estimating the likelihood of a region R: A region R is represented as an *ensemble of descriptors* $\{d_i\}$, with their relative positions $\{l_i\}$ within R. Let $p(R|Ref, T)$ denote the likelihood to find the region $R \subset Q$ in another image Ref at a location/transformation denoted by T. This likelihood is estimated by the similarity between the descriptors of R and the corresponding descriptors (according to T) in Ref:

$$p(R|Ref, T) = \frac{1}{Z} \prod_i \exp - \frac{|\Delta d_i(Ref, T)|^2}{2\sigma^2} \quad (1)$$

where $\Delta d_i(Ref, T)$ is the error between the descriptor $d_i \in R$ and its corresponding descriptor (via T) in Ref and Z is a normalization factor. We use the following

approximation of the likelihood of R, $p(R|Ref)$ according to its best match in Ref:

$$p(R|Ref) \triangleq \max_{T} p(R|Ref, T)p(T) \tag{2}$$

(This forms a lower bound on the true likelihood)

In our current implementation, the descriptors $\{d_i\}$ were chosen to be two types of descriptors (estimated densely in the image): HOG [11] and Local Self-Similarity [12]. These two descriptors have complementary properties: the first captures local texture information, whereas the second captures local shape information while being invariant to texture (e.g., different clothing). We assume a uniform prior $p(T)$ on the transformations T over all pure shifts. We further allow small local non-rigid deformations of R (slight deviations from the expected (relative) positions $\{l_i\}$ of $\{d_i\}$). Scale invariance is introduced separately (see Sec. 5).

The 'Statistical Significance' of a region R: Recall that we wish to detect *large non-trivial* recurring regions across images. However, the *larger* the region, the *smaller* its likelihood according to Eq. (2). In fact, tiny uniform regions have the highest likelihood (since they have lots of good matches in Ref). Thus, it is not enough for a region to match well, but should also have a low probability to occur at random, i.e.:

$$Likelihood\ Ratio\ (R) = \frac{p(R|Ref)}{p(R|H_0)} \tag{3}$$

This is the likelihood ratio between the probability of generating R from Ref, vs. the probability of generating R *at random* (from a "random process" H_0). $p(R|H_0)$ measures the statistical ***in***significance of a region (high probability = low significance). If a region matches well, but is trivial, then its likelihood ratio will be low (inducing a low affinity). On the other hand, if a region is non-trivial, yet has a good match in another image, its likelihood ratio will be high (inducing a high affinity).

We next present an approach we developed for efficiently estimating $p(R|H_0)$, i.e. the chance of a region R to be generated at random. Assuming descriptors $d_i \in R$ are independent: $p(R|H_0) = \prod_i p(d_i|H_0)$. Given a set of images (the images we wish to cluster, or a general set of natural images), we define D to be the collection of all the descriptors extracted from those images. We define $p(d|H_0)$ to be the probability of randomly sampling the descriptor d from the collection D (or its frequency in D). This can be estimated using Parzen density estimation, but is too time consuming. Instead, we quantize D into a small rough 'codebook' $\hat{D}$ of a few hundred codewords (e.g., using k-means or even just uniform sampling). Frequent descriptors in D will be represented well in $\hat{D}$ (have low quantization error relative to their nearest codeword), whereas rare descriptors will have high quantization error. This leads to the following rough approximation, which suffices for our purpose: $p(d|H_0) = \exp -\frac{|\Delta d(H_0)|^2}{2\sigma^2}$, where $\Delta d(H_0)$ is the error between d and its most similar codeword in $\hat{D}$.

Fig. 3 displays $\Delta d(H_0) \propto -\log p(d|H_0)$ for a few images of the Ballet/Yoga and the Animals datasets. Red marks descriptors (HOG) with high error $\Delta d(H_0)$, i.e., high statistical significance. Image regions R containing many such descriptors have high statistical significance (low $p(R|H_0)$). Statistically significant regions in Fig. 3.a appear to coincide with body gestures that are unique and informative to the separation

between Ballet and Yoga. Recurrence of such regions across images will induce strong and reliable affinities for clustering. Observe also that the long horizontal edges (between the ground and sky in the Yoga image, or between the floor and wall in the Ballet images) are not statistically significant, since such edges occur abundantly in many images. Similarly, statistically significant regions in Fig. 3.b coincide with parts of the animals that are unique and informative for their separation (e.g., the Monkey's face and hands, the Elk's horns, etc.) This is similar to the observation of [13] that the most informative descriptors for classification tend to have the highest quantization error.

Unlike the common use of codebooks ("bags of descriptors") in recognition, here the codebook is NOT used for representing the images. On the contrary, a descriptor which appears frequently in the codebook is "ignored" or gets very low weight, since it is very frequently found in the image collection and thus not informative.

The "Saving in Bits" obtained by a region R: According to Shannon, the number of bits required to 'code' a random variable x is $-\log p(x)$. Taking the log of Eq. (3) and using the quantized codebook $\hat{D}$ yields (disregarding global constants):

$$\log \frac{p(R|Ref)}{p(R|H_0)} = \sum_i |\Delta d_i(H_0)|^2 - |\Delta d_i(Ref)|^2 = \textbf{"savings in bits"} \qquad (4)$$

This is the number of bits saved by generating R from Ref, as opposed to generating it 'from scratch' at random (using H_0). Therefore, if a region R is composed of statistically significant descriptors (with high $\Delta d_i(H_0)$), and has a good match in Ref (low $\Delta d_i(Ref)$), then R will obtain very high 'savings in bits' (because the difference between the two errors is large). In contrast, a large recurring uniform region or a long edge will hardly yield any 'savings in bits', since both errors $\Delta d_i(H_0)$ and $\Delta d_i(Ref)$ will be low, resulting in a small difference.

So far we discussed a single region R. When the query image Q is composed of multiple (non-overlapping) regions $R_1, .., R_r$ from Ref, we approximate the *total* 'savings in bits' of Q given Ref, by summing up the 'savings in bits' of the individual regions. This forms the affinity between Q and Ref:

$$\text{affinity(Q,Ref)} = savings(Q|Ref) = \sum_{i=1}^{r} savings(R_i|Ref) \qquad (5)$$

4 Randomized Detection of Shared Regions

We propose a randomized algorithm for detecting large unknown (irregularly shaped) regions which are shared across images. Inspired by "PatchMatch" [10], we exploit the power of random sampling and the coherence of neighboring pixels (descriptors) to quickly propagate information. However, unlike PatchMatch, we search for 'matching *regions*', as opposed to matching patches/descriptors.

Although efficient, the theoretical complexity of PatchMatch is $O(N \log N)$, where N is the size of the images. This is because it spends most of its time seeking good matches for the spurious and isolated descriptors. However, in our application, we wish

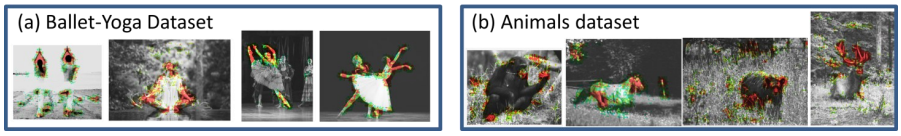

Fig. 3. Statistical significance of descriptors. *Red signifies HOG descriptors with the highest statistical significance (descriptors that rarely appear). Green – lower significance; Grayscale – much lower.*

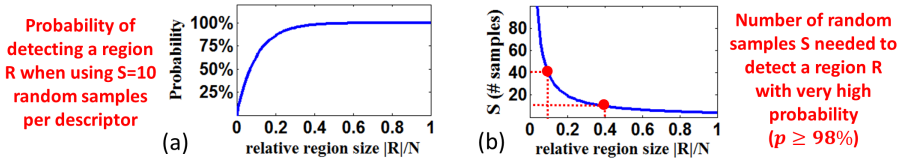

Fig. 4. Illustration of Claim 1

to find only *large* matching regions across images (and ignore the small spurious distracting ones). We show that this can be done in linear time $O(N)$. In fact, *the larger the region, the faster it will be found* (with fewer random samples, and with higher probability). In other words, and quite surprisingly: *region-matching is 'easier' than descriptor-matching!* We refer to this as the "wisdom of crowds of pixels".

The Region Growing (Detection) Algorithm:

Let R be a shared region (of unknown shape, size, or position) between images I_1 and I_2 of size N. Let R_1 and R_2 denote its instances in I_1 and I_2, respectively. The goal is to find for each descriptor $d_1 \in R_1$ its matching descriptor $d_2 \in R_2$.

(i) Sampling: Each descriptor $d \in I_1$ *randomly* samples S locations in I_2, and chooses the best one. The complexity of this step is $O(SN)$. The chance of a single descriptor d to accidently fall on its correct match in I_2 is very small. However, the chance that *at least one* descriptor from R_1 will accidently fall on its correct match in R_2 is very high if R is large enough (see Claim 1 and Fig. 4.a.). Once a descriptor from R_1 finds a good match in R_2, it propagates this information to all the descriptors in R_1:

(ii) Propagation: Each descriptor chooses between its best match, and the match proposed by its spatial neighbors. This is achieved quickly via two image sweeps (once from top down, and once from bottom up). The complexity of this step is $O(N)$.

Complexity: The overall runtime is $O(SN)$. In Claim 1, we prove that for large enough regions, the required S is a small constant, yielding a overall linear complexity, $O(N)$.

Next we provide a series of claims (Claims 1-4) which quantify the number of random samples per descriptor S required to detect shared regions R across images (*pairs* and *collections* of images) at high probability. This is analyzed as a function of the relative region size in the image $|R|/N$, the desired detection probability p, and the number of images M in the image collection. **All proofs appear on our project website** `www.wisdom.weizmann.ac.il/~vision/ClusterByComposition.html`

4.1 Shared Regions between Two Images

For the purpose of analysis only, we assume that the size of all images is N, and that the transformation between shared regions is a pure rigid shift.

Claim 1 [A single shared region between two images]

Let R be a region which is shared by two images, I_1 and I_2 of size N. Then:
(a) Using S random samples per descriptor, guarantees to detect the region R with probability $p \geq \left(1 - e^{-S\,|R|/N}\right)$.
(b) To guarantee the detection of the region R with probability $p \geq (1-\delta)$, requires $S = \frac{N}{|R|}\log(\frac{1}{\delta})$ random samples per descriptor.

Proof: See **project website** ■

Implication: Fig. 4.a,b graphically illustrates the terms in claim 1.a,b. For example, to detect a shared region of relative size 10% with probability $p \geq 98\%$ requires $S = 40$. Thus, a complexity of $O(40N)$ – linear in N.

Claim 2 [Multiple shared regions between two images]

Let $R_1, \ldots, R_L$ be L shared non overlapping regions between two images I_1 and I_2. If $|R_1| + |R_2| + \ldots |R_L| = |R|$, then it is guaranteed to detect at least one of the regions R_i with the same probability $p \geq (1-\delta)$ and using the same number of random samples per descriptor S as in the case of a single shared region of size $|R|$.

Proof: See **project website** ■

Implication: Consider the case where at least 40% of one image can be composed using several (smaller) pieces of another image. Then according to Fig. 4.b, when using only $S = 10$, we are guaranteed to detect at least one of the shared regions with probability $p \geq 98\%$. Moreover, as shown in Fig. 4.a, this region will most likely be one of the largest regions in the composition, since small regions have very low detection probability with $S = 10$ (e.g., a region of size 1% has only 10% chance of detection).

4.2 Shared Regions within an Image Collection

We now consider the case of detecting a shared region between a query image and at least one other image in a large collection of M images. For simplicity, let us first examine the case where all the images in the collection are "partially similar" to the query image. We say that two images are **"partial similar"** if they share at least one large region (say, at least 10% of the image size). The shared regions R_i between the query image and each image I_i in the collection may be possibly different ($R_i \neq R_j$).

Claim 3 [Shared regions within an image collection]

Let I_0 be a query image and let $I_1 ... I_M$ be "partially similar" to I_0. Let $R_1 ... R_M$ be regions of size $|R_i| \geq \alpha N$ such that R_i is shared by I_0 and I_i (regions R_i may overlap in I_0). Using $S = \frac{1}{\alpha}\log(\frac{1}{\delta})$ samples per descriptor in I_0, distributed randomly across $I_1 ... I_M$, guarantees with probability $p \geq (1-\delta)$ to detect at least one region R_i.

Proof: See **project website** ■

Implication: The above claim entails that using the *same number of samples* S per descriptor as in the case of two images, but now *scattered randomly across the entire image collection,* we are still guaranteed to detect at least one of the shared regions with high probability. This is **regardless of the number of images *M* in the collection!** For example, if the regions are at least 10% of the image size (i.e., $\alpha = 0.1$), then $S = 40$ random samples per descriptor in I_0, distributed randomly across $I_1, .., I_M$, suffice to detect at least one region R_i with probability 98%.

In practice, however, only a portion of the images in the collection are "partially similar" to I_0, and those are 'buried' among many other non-similar images. Let the number of "partially similar" images be $\frac{M}{C}$, where $\frac{1}{C}$ is their portion in the collection. It is easy to show that in this case we need to use C times more samples in order to find at least one shared region between I_0 and one of the "partially similar" images. For example, assuming there are 4 clusters and assuming all images in the cluster are "partially similar" (which is not always the case) then $C = 4$. Note that typically, the number of clusters is much smaller than the number of images, i.e. $C \ll M$.

Claim 4 [Multiple images vs. Multiple images]
Assume each image in the collection is "partially similar" with at least $\frac{M}{C}$ *images (shared regions* $\geq 10\%$ *image size), and* $S = 40C$ *samples per descriptor. Then one 'simultaneous iteration' (all the images against each other), guarantees that at least* 95% *of the images will generate at least one strong connection (find a large shared region) with at least one other image in the collection with high probability. This probability rapidly grows with the number of images, and is practically* 100% *for* $M \geq 500$.

Proof: See **project website** ■

Implication: Very few iterations thus suffice for all images to generate at least one strong connection to another image in the collection. Fig. 5 shows a few examples of such connecting regions detected by our algorithm in the Ballet/Yoga dataset.

5 The Collaborative Image Clustering Algorithm

So far, each image independently detected its own shared regions within the collection, using only its 'internal wisdom' (the "wisdom of *crowds of pixels*"). We next show how **collaboration between images** can significantly improve this process. Each image can further make *'scholarly suggestions'* to other images where they should sample and search within the collection. For example, looking at Fig. 1, image I_0 has strong affinity to images $I_1, .., I_4$. Therefore, in the next iteration, I_0 can 'encourage' $I_1, .., I_4$ to search for matching regions in each other. Thus, e.g., I_3 will be 'encouraged' to sample more in I_1 in the next iteration. The guided sampling process via multi-image collaboration significantly speeds up the process, reducing the required number of random samples and iterations. Within few iterations, strong connections are generated among images belonging to the same cluster. We refer to this as the "wisdom of *crowds of images*".

Instead of exhausting all the random samples at once, the region detection is performed iteratively, each time using a small number of samples. Most of the detected

regions in the first iterations are inevitably large, since only large regions have high enough probability to be detected given a small number of samples (according to Claims 1-2 and Fig. 4.a). Subsequently, as we perform more iterations, more samples are added, and smaller regions also emerge. However, the detection of smaller shared regions is guided by the connections already made (via the larger shared regions). This increases the chance that these small regions are also meaningful, thus strengthening connections within the cluster, rather than detecting small distracting coincidental regions. We typically choose S (the number of random samples per descriptor), to be twice the number of clusters (at each iteration). For example, for 4 clusters $S = 8$ at each iteration.

In a nut-shell, our algorithm starts with *uniform* random sampling across the entire collection. The connections created between images (via detected shared regions) induce affinities between images (see Sec. 3). At each iteration, the *sampling density distribution* of each image is re-estimated according to 'suggestions' made by other images (guiding it where to sample in the next iteration). This results in a "guided" random walk through the image collection. Finally, the resulting affinities (from all iterations) are fed to the N-Cut algorithm [14], to obtain the desired clusters.

Note that N-Cut algorithm (and other graph partitioning algorithms) implicitly rely on two assumptions: (i) that there are enough strong affinities within each cluster, and (ii) that the affinity matrix is relatively sparse (with the hope that there are not too many connections across clusters). The sparsity assumption is important both for computational reasons, as well as to guarantee the quality of the clustering. This is often obtained by sparsifying the affinity matrix (e.g., by keeping only the top $10 \log_{10} M$ values in each row [6]). The advantage of our algorithm is that it implicitly achieves both conditions via the 'scholarly' multi-image collaborative search. The 'suggestions' made by images to each other quickly generate (within a few iterations) strong intra-cluster connections, and very few inter-cluster connections.

Notations:

• F_i denotes the mapping between the descriptors of image I_i to their matching descriptors in the image collection. It contains for each descriptor the index of the image of its match and its spatial displacement in that image. F_i tends to be piece-wise smooth in areas where matching regions were detected, and quite chaotic elsewhere.

• A denotes the affinity matrix of the image collection. Our algorithm constructs this matrix using the information obtained by composing images from each other.

• B denotes the"Bit-Saving" matrix at each iteration. The value B_{ij} is "Saving in Bits" contributed by image I_j to the composition of image I_i, at a specific iteration.

• P denotes the "sampling distribution" matrix at each iteration. P_{ij} is the prior probability of a descriptor in image I_i to randomly sample descriptors in image I_j when searching for a new candidate match. P_i (the i_{th} row of P) determines how image I_i will distribute its samples across all other images in the collection in the next iteration.

• U denotes the matrix corresponding to a *uniform* sampling distribution across images. (i.e., all its entries equal $\frac{1}{M-1}$, except for zeros on the diagonal).

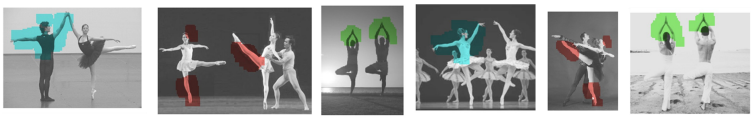

Fig. 5. Examples of shared regions detected by our algorithm. *Detected connecting regions across images are marked by the same color.*

The algorithm;

Initiate affinity matrix to zero: $A \equiv 0$ *and sampling distributions to uniform:* $P = U$;
for *iteration* $t = 1, \ldots, T$ **do**
  for *image* $i = 1, \ldots, M$ **do**
    1. Randomly sample according to distribution P_i using S samples;
    2. 'Grow' regions (Sec. 4), allowing small non-rigidities. This results in the mapping F_i;
    3. Update row i of B ("Bit-Saving") according to mapping F_i;
  end
  Update affinity matrix: $A = \max(A, B)$;
  Update P (using the "wisdom of crowds of images");
  if $(t \bmod J) = 0$ (i.e every J-th iteration): Reset the mappings $F_1, ..., F_M$;
end
Impose symmetry on the affinity matrix A by $A = max(A, A^T)$;
Apply N-cut [14] on A to obtain K image clusters (we assume K is known);

Explanations:

• **Randomly sample according to distribution P_i:** Each descriptor in image I_i samples descriptors at S random locations in the image collection. Each of the S samples is sampled in 2 steps: (i) an image index $j = 1, \ldots, M$ is sampled according to distribution P_i (ii) a candidate location in I_j is sampled uniformly. If one of the new candidates for this descriptor improves the current best match, then it becomes the new match.

• **Allow small non-rigidities:** We add a local refinement sampling phase at the vicinity of the current best match, thus allowing for small non-rigid deformations of matched regions (implemented similarly to the local refinement phase of [10]).

• **Update "Bit-Savings" matrix B:** The detected ("grown") regions need *not* be explicitly extracted in order to compute the "Savings-in-bits". Instead, for each image I_i, we first disregard all the descriptors which are spuriously mapped by F_i (i.e mapped in an inconsistent way to their surrounding). Let χ_i denote all remaining descriptors in image I_i. These descriptors are part of larger regions grown in I_i. $B(i,j)$ is estimated using the individual pixel-wise "Savings-in-bits" induced by the mapping F_i, summed over all the descriptors in χ_i which are mapped to image I_j:
$B(i,j) = \sum_{k \in \chi, F_i(k) \mapsto I_j} |\Delta d_k(H_0)|^2 - |\Delta d_k(I_j)|^2$, where $\Delta d_k(I_j)$ is the error between descriptor d_k in I_i and its match in I_j (induced by F_i).

• **Update P (using the "wisdom of *crowds of images*"):** Let us consider a Markov chain (a "Random Walk") on the graph whose nodes are the images in the collection.

We set the transition probability matrix between nodes (images) $\hat{B}$ to be equal to the "Bit-Savings" matrix B, after normalizing each row to 1. $\hat{B}(i,j)$ reflects the *relative* contribution of each image to the current composition of image I_i. If we start from state i (image I_i) and go one step in the graph, we will get a distribution equal to $\hat{B}_i$ (the image own "wisdom"). Similarly, if we go two steps we get a distribution $\hat{B}_i^2$ (the neighbors' "wisdom"). Using these facts, we update the sampling distributions in P as follows: $P = w_1\hat{B} + w_2\hat{B}^2 + w_3U$, where $w_1 + w_2 + w_3 = 1$. The first term, $\hat{B}$, encourages each image to keep sampling in those images where it already found initial good regions. The second term, $\hat{B}^2$, contains the 'scholarly' suggestions that images make to each other. For example, if image I_i found a good region in image I_j (high $\hat{B}_{ij}$), and image I_j found a good region in image I_k (high $\hat{B}_{jk}$), then $\hat{B}^2_{ik}$ will be high, suggesting that I_i should sample more densely in image I_k in the next iteration. The third term, U, promotes searching uniformly in the collection, to avoid getting 'stuck' in local minima. The weights w_1, w_2, w_3 gradually change with the J internal iterations. At the beginning more weight is given to uniform sampling and at the end of the process more weight is given to to the guided sampling.

• **Reset the mappings $\{F_i\}$:** This is done every few iterations ($J = 3$) in order to encourage the images to restart their search in other images and look for new connections.

Incorporating Scale Invariance: In order to handle scale invariance, we generate from each image a cascade of multi-scale images, with relative scales $\{(\sqrt{0.5})^l\}_{l=0}^{3}$ - images of size $\{1, 0.7, 0.5, 0.35\}$ relative to the original image size (in each dimension). The region detection algorithm is applied to the entire multi-scale collection of images, allowing region growing also across different scales between images. The multi-scale cascade of images originating from the same input image are associated with the same entity in the affinity matrix A.

Complexity (Time & Memory): All matrix computations and updates ($max(A, B)$, $\hat{B}^2$, update P, etc.) are efficient, both in terms of memory and computation, since the matrix B is *sparse*. Its only non-zero entries correspond to the image connections generated in the current iteration. We set the number of iterations to be $T = 10\log_{10} M$, which is the recommended sparsity of the affinity matrix by [6]. Note that our algorithm directly estimates a good set of sparse affinities (as opposed to computing a full affinity matrix and then sparsifying it). T is typically a small number (e.g., for $M = 1000$ images $T = 30$; for $M = 10,000$ images $T = 40$). The complexity of each iteration is $O(NM)$ (see Sec. 4). Therefore, the overall complexity of our clustering algorithm is $O(NMlog_{10}(M))$ - almost linear in the size of the image collection (NM).

6 Experimental Results

We tested our algorithm on various datasets, ranging from evaluation datasets (Caltech, ETHZ), on which we compared results to others, to more difficult datasets (PASCAL), on which to-date *no* results were reported for *purely unsupervised* category discovery. Finally, we also show the power of our algorithm on *tiny* datasets. Tiny datasets are challenging for *unsupervised* learning, since there are very few images to 'learn' from.

Table 1. Performance evaluation on benchmark datasets

Benchmark	# of classes	Measure	[3]	[4]	[7]	Our Method	Our Method (with restricted search range)
Caltech	4	F-measure	-	-	0.87	**0.89**	**0.96**
Caltech	10	F-measure	-	-	0.68	**0.79**	**0.87**
Caltech	7	Purity	-	-	78.9	**89.8**	**90**
Caltech	20	Purity	-	-	65.6	**78.9**	**86.3**
ETHZ	5	Purity	76.5	87.3	-	**89**	**95.3**

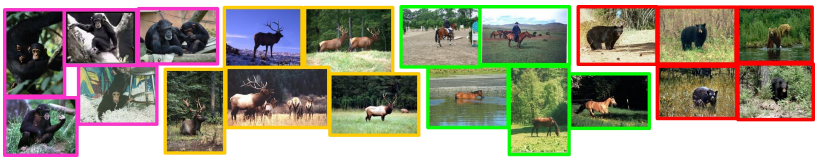

Fig. 6. Clustering Results on our Animal dataset (horses, elks, chimps, bears).

Experiments on Benchmark Evaluation Datasets: We used existing benchmark evaluation datasets (Caltech, ETHZ-shape) to compare results against ([7],[3],[4]) using their experimental setting and measures. Results are reported in Table 1. The four datasets generated by [7] consist of difficult classes from Caltech-101 with non-rigid objects and cluttery background (such as leopards and hedgehogs), from 4 classes (189 images) up to 20 classes (1230 images). ETHZ consists of 5 classes: Applelogos, Bottles, Giraffes, Mugs and Swans. For the ETHZ dataset, we followed the experimental setting of [3] (which crops the images so that the objects are 25% of the image size). For both Benchmarks, our algorithm obtains state-of-the-art results (see Table 1). Furthermore, when *restricting the spatial search range* of descriptors to no more than 25% of the image size (around each descriptor), results improve significantly (see Table 1). Such a restriction enforces a weak prior on the rough geometric arrangement within the image. Note that for the case of 10 and 20 Caltech classes, our algorithm *obtains* 30% *relative improvement over current state-of-the-art*.

Experiments on a Subset of Pascal-VOC 2010 Dataset: The Pascal dataset is a very challenging dataset, due to the large variability in object scale, appearance, and due to the large amount of distracting background clutter. Unsupervised category discovery is a much more difficult and ill-posed problem than classification, therefore to-date, *no* results were reported on PASCAL for *purely unsupervised* category discovery. We make a first such attempt, restricting ourselves at this point to 4 categories: Car, Bicycle, Horse and Chair. We generated a subset of 100 images per category restricting ourselves to images labeled "side view" and removing images which simultaneously contain objects from 2 or more of the above categories (otherwise the clustering problem is not well-defined). See resulting subset in our **project website** `www.wisdom.weizmann.ac.il/~vision/ClusterByComposition.html`. Note that in many of the images the object is extremely small. We tested our algorithm on this subset and obtained a mean

purity of 66.5% (a 20% relative improvement over a bag-of-words + N-cut baseline). In this case, restricting the search range did not yield better results. More detailed clustering results of the PASCAL subset and their analysis appear in the **project website**.

Experiments on *Tiny* Datasets: Existing methods for unsupervised category discovery require a *large* number of images per category (especially for complex non-rigid objects), in order to 'learn' shared 'cluster models'. To further show the power of our algorithm, we generated two *tiny* datasets: the Ballet-Yoga dataset (Fig. 2) and the Animal dataset (Fig. 6). These tiny datasets are *very challenging* for unsupervised category discovery methods, because of their *large* variability in appearance versus their *small* number of images. Our algorithm obtains excellent clustering results for both datasets, even though each category contains different poses, occlusions, foreground clutter (e.g. different clothes), and confusing background clutter (e.g. in the animal dataset). The success of our algorithm can be understood from Figs. 3, 5: Fig. 3 shows that the descriptors with the highest statistical significance are indeed the most informative ones in each category (e.g., the Monkey's face and hands, the Elk's horns, etc.). Fig. 5 shows that meaningful shared regions were detected between images of the same category.

References

1. Grauman, K., Darrell, T.: Unsupervised learning of categories from sets of partially matching image features. In: CVPR. (2006)
2. Russell, B.C., Efros, A.A., Sivic, J., Freeman, W.T., Zisserman, A.: Using multiple segmentations to discover objects and their extent in image collections. In: CVPR. (2006)
3. Lee, Y.J., Grauman, K.: Shape discovery from unlabeled image collections. In: CVPR. (2009)
4. Payet, N., Todorovic, S.: From a set of shapes to object discovery. In: ECCV. (2010) 57–70
5. Sivic, J., Russell, B., Efros, A., Zisserman, A., Freeman, W.: Discovering objects and their localization in images. In: ICCV. (2005)
6. Kim, G., Faloutsos, C., M.Hebert: Unsupervised modeling of object categories using link analysis techniques. In: CVPR. (2008)
7. Lee, Y.J., Grauman, K.: Foreground focus: Unsupervised learning from partially matching images. IJCV **85** (2009) 143–166
8. Boiman, O., Irani, M.: Similarity by composition. In: NIPS. (2006)
9. Gu, C., Lim, J.J., Arbelaez, P., Malik, J.: Recognition using regions. In: CVPR. (2009)
10. Barnes, C., Shechtman, E., Finkelstein, A., Goldman, D.B.: Patchmatch: A randomized correspondence algorithm for structural image editing. In: SIGGRAPH. (2009)
11. Dalal, N., Triggs, B.: Histograms of oriented gradients for human detection. In: CVPR. (2005)
12. Shechtman, E., Irani, M.: Matching local self-similarities across images and videos. In: CVPR. (2007)
13. Boiman, O., Shechtman, E., Irani, M.: In defense of nearest-neighbor based image classification. In: CVPR. (2008)
14. Shi, J., Malik, J.: Normalized cuts and image segmentation. TPAMI **22** (2000) 888–905