Foreground Focus: Finding Meaningful Features in Unlabeled Images

Yong Jae Lee and Kristen Grauman
University of Texas at Austin
yjlee0222@mail.utexas.edu, grauman@cs.utexas.edu

Abstract

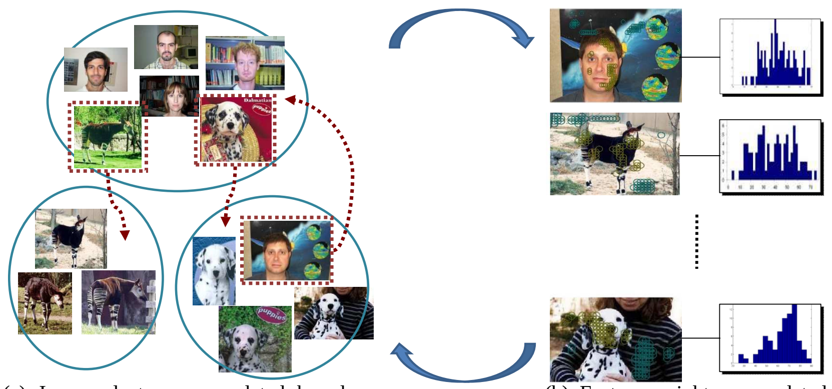
We present a method to automatically discover meaningful features in unlabeled image collections. Each image is decomposed into semi-local features that describe neighborhood appearance and geometry. The goal is to determine for each image which of these parts are most relevant, given the image content in the remainder of the collection. Our method first computes an initial image-level grouping based on feature correspondences, and then iteratively refines cluster assignments based on the evolving intra-cluster pattern of local matches. As a result, the significance attributed to each feature influences an image's cluster membership, while related images in a cluster affect the estimated significance of their features. We show that this mutual reinforcement of object-level and feature-level similarity improves unsupervised image clustering, and apply the technique to automatically discover categories and foreground regions in images from benchmark datasets.

1 Introduction

Learning to describe and recognize visual objects is a fundamental problem in computer vision that serves as a building block to many potential applications. Recent years have shown encouraging progress, particularly in terms of generic visual category learning [25, 12, 26, 2, 13] and robust local feature representations [16, 1, 10]. A widespread strategy is to determine the commonalities in appearance and shape amongst a group of labeled images, and then search for similar instances in new images based on those patterns. Typically a supervised setting is assumed, where the recognition method is trained with manually prepared exemplars of each class of interest.

However, carefully labeled exemplars are expensive to obtain in the large numbers needed to fully represent a category's variability, and methods trained in this manner can suffer from unintentional biases imparted by dataset creators. Recognition methods stand to gain from stores of unstructured, unlabeled images and videos, if they can infer which basic visual patterns are meaningful. While recent work has begun to address the need for looser supervision requirements [25, 26, 6, 8], learning from completely unlabeled images remains difficult. Unsupervised learners face the same issues that plague supervised methods—clutter, viewpoint, intra-class appearance variation, occlusions—but must handle them without any explicit annotation guidance.

We are interested in automatically identifying the foreground object(s) of interest among an unlabeled pool of images. To qualify as foreground, we say that the visual


(a) Image clusters are updated based on weighted semi-local feature matches.

(b) Feature weights are updated based on current clusters.

Figure 1: Illustration of the proposed method. The images are grouped based on weighted semi-local feature matchings (a), and then image-specific feature weights are adjusted based on their contribution in the match relative to all other intra-cluster images (b). These two processes are iterated (as denoted by the block arrows in the center) to simultaneously determine foreground features while improving cluster quality. Dotted arrows denote images with updated cluster memberships.

pattern must have observable support within the collection—that is, it must re-occur repeatedly, albeit with some variation in appearance across the instances. Isolating "important" features that are responsible for generating natural image clusters would be useful to detect discovered objects in novel images, or to generate compact summaries of visual content. Thus the task is essentially unsupervised feature subset selection: to determine which portion of the features present can be used to form high quality clusters under a chosen clustering objective.

We propose a solution that seeks the mutual support between discovered objects and their defining features. An initial image-level grouping is computed based on the correspondences between any two images' semi-local features. Within each initial group, the pattern of the matches is analyzed to determine how responsible each part was for placing its parent image into its current cluster. From this, we compute a significance weight on each feature. The groups and feature weights are then iteratively refined by alternately computing 1) the cluster membership given the re-weighted features and 2) the feature weights given the newly refined memberships (see Figure 1). Due to the reciprocal reinforcement, the iterative process yields both a partition of the unlabeled inputs as well as their detected foreground. We also define a new semi-local region descriptor to provide a flexible encoding of local appearance and geometry.

Our results show that unsupervised foreground feature detection aids in grouping similar objects, while important features are better found on objects of interest when given partitions of partially re-occurring patterns. We compare our approach with existing unsupervised learning algorithms and show improvements on benchmark datasets.

2 Related Work

In this section we review relevant work in supervised image feature selection, weakly supervised and unsupervised category learning, and semi-local descriptors.

Various recognition methods can learn categories from labeled images with segmented foreground and then detect them within cluttered images; in [12, 17], the authors show how to weight features matched to a novel test image based on their agreement with known object geometry, thereby downplaying background. The paradigm of "weak supervision" suggested in [25] has since been pursued by a number of methods (e.g. [26, 2]). In this model, categories are learned from cluttered, unsegmented labeled images; one seeks the parts that best fit all examples sharing the same label. Discriminative feature selection strategies have also been explored to detect features that occur frequently in in-class examples but rarely on the background [18, 3]. The idea of "cosegmentation" was introduced in [20], where the common parts of an image pair are simultaneously segmented. Our approach shares the goal of identifying consistent features in cluttered images, but unlike the above methods it does not employ any labeled examples to do so.

Recent work has considered ways to discover latent visual themes in images using topic models developed for text, such as probabilistic Latent Semantic Analysis (pLSA) or Latent Dirichlet Allocation [19, 22, 6, 15]. The models use feature co-occurrence patterns to recover the underlying distributions (topics) that best account for the data. The notion of text documents containing unordered words can be transferred to images composed of "visual words". Recent extensions incorporate spatial constraints [6, 15], or use segmentation to reduce the spatial extent of each "document" [22].

Other approaches treat unsupervised category learning as an image clustering problem. In [8], local feature matches are used with spectral clustering, and in [4] a message-passing algorithm propagates non-metric affinities and identifies good exemplars. Our method also begins by computing pairwise affinities between images. In contrast to these techniques, however, the proposed approach allows common feature matches to reinforce and refine the discovered groups; as a result it provides both the groupings as well as the predicted foreground-background separation.

The problem of unsupervised feature selection has received limited attention in the machine learning community (see [5] and references therein), but existing methods presume a vector input space, many assume the data to be generated by certain parametric distributions, and/or are specifically tailored to a particular clustering method—any of which can be ill-suited for the visual learning scenario.

Researchers have proposed "semi-local" feature descriptors that capture information about local neighborhoods surrounding an interest point [10, 1, 18, 24]. The general idea is to build more specific features that reflect some geometry and aggregate nearby features into a single descriptor. In order to compute more reliable correspondences between images, we design a new descriptor that counts the co-occurrence of each visual word type relative to an interest point, accumulating the counts at increasingly distant spatial regions and in distinct relative configurations.

Our main contribution is a new approach to perform unsupervised foreground feature selection from collections of unlabeled images. Whereas previous feature selection methods could detect foreground or discriminative features in labeled images, our method discovers them in unlabeled images. In practice, we show that this allows more accurate unsupervised category learning with benchmark recognition datasets.

3 Approach

The goal is to predict which regions in unlabeled images correspond to foreground, and in doing so to improve accuracy in unsupervised visual pattern discovery. Given a set of unlabeled images, our method groups similar examples based on the correspondence between their semi-local features. After an initial grouping, we weight each feature according to its contribution to the match between the image that contains it and every other intra-cluster image. Then, the groupings and weights for the whole image collection are iteratively re-computed, in the end producing both a partition of the image collection as well as weights that reflect the degree to which a feature is believed to be foreground. We first describe the grouping process in detail, and then overview our semi-local descriptor.

3.1 Simultaneous Image Grouping and Foreground Detection

Given N unlabeled images, $U = \{I_1, ..., I_N\}$, we represent each image I_i as a set of weighted features, $X_i = \{(f_1, w_1), (f_2, w_2), ..., (f_{|X_i|}, w_{|X_i|})\}$, where each $f_j \in \Re^d$ is a local image descriptor weighted with some $w_j \geq 0$. The weight on a feature vector determines its importance within the image, and will affect any matching computed for the set in which it is contained. Initially, all feature weights are set to a uniform value (one). Subsequently, every time we cluster the images, the support (or lack of support) computed for a feature within a group will result in an increase (or decrease) of its weight. Those weight updates in turn influence the image groups found at the next iteration.

Clustering Weighted Feature Sets. A good clustering should group together images that have a consistent repeated appearance pattern. However, given that the images will likely be cluttered and may contain multiple objects, the pattern need not encompass the entire image. Therefore, we want to compute clusters based on the appearance agreement of some portion of each example—that is, based on a match between subsets of the semi-local features. Further, the weight on a feature should dictate how much attention an image-to-image comparison pays to it, so that features with high weight have more influence on the measured cost of a match, and features with low weight have little effect.

To accomplish such a grouping, we perform spectral clustering with an affinity matrix that reflects the least-cost partial matching between weighted point sets. Also known as the Earth Mover's Distance (EMD) [21], this optimal match cost $M(X,Y)$ reflects how much effort is required to transform weighted point set X into weighted point set Y:

$$M(X,Y) = \frac{\sum_i \sum_j F_{i,j} \;\; D(X(f_i), Y(f_j))}{\sum_i \sum_j F_{i,j}}, \tag{1}$$

where $X(f)$ and $Y(f)$ denote features from sets X and Y, respectively, and $D(X(f_i), Y(f_j))$ denotes the distance between points $X(f_i)$ and $Y(f_j)$. The values $F_{i,j}$ are scalars giving the *flow*, or amount of weight that is mapped from point $X(f_i)$ to point $Y(f_j)$. Note that this takes into account the distance between matched points as well as the amount of weight (mass or "dirt") attached to each one. The EMD has previously been used in supervised tasks to compare textures' local feature distributions [9].

In our case, we use the weights to encode priority in the matching: assuming an image's foreground features are relatively highly weighted, a second image cannot produce a low matching cost against it unless it has similar point(s) to the foreground with similar total weight(s). Likewise, a feature with low weight cannot contribute much cost to any match, so its influence is negligible. At each clustering iteration, we compute affinities using the $N \times N$ matrix $\mathbf{C}$ of matching scores between all pairs of unlabeled images: $\mathbf{C}_{m,n} = M(X_m, X_n)$, for $m, n = 1, \ldots, N$. These affinities are input to a spectral clustering algorithm that partitions the N examples into k groups. In our implementation we use the normalized cuts criterion [23], due both to its efficiency and the fact that it prefers farther-reaching clusters; however alternate spectral methods are plausible.

Refining Foreground Feature Weights from the Current Clusters. Given a k-way partition of the images, we update the weights attached to each feature by leveraging any current regions of agreement among the images in a single partition. Even when all pairs of examples within a cluster have high matching similarity, because each matching can draw from different combinations of features, heterogenous clusters are possible. To overcome this, we look to the pattern of the flow fields computed by Eqn. 1. The idea is to use information among the "good" matches (images amongst which all pairs have similar matching points) to re-interpret the "bad" matches (images amongst which similar matching points exist, but are not consistent across all intra-cluster pairs).

The flow field specifies which features in two images best match which, and using what amount of weight. Given a cluster of C images $\{X_1,\ldots,X_C\}$, for each example X_i, $i = 1,\ldots,C$, we define $(C-1)$ $|X_i|$-dimensional weight vectors denoted $\mathbf{w}_{ij}$, with $j = \{\{1,\ldots,C\} \setminus i\}$. That is, we compute a vector of feature weights for the i-th example against every other image within the cluster. Each of the weight entries in $\mathbf{w}_{ij}$ specifies how much its feature from X_i contributed to the match with set X_j. We define the d-th element as $\mathbf{w}_{ij}(d) = \sum_{p=1}^{|X_j|} F_{d,p}/D(X_i(f_d),X_j(f_p))$, for $d = 1,\ldots,|X_i|$. Each weight is the sum of all the flow amounts from that particular feature in X_i to any other feature in the other set X_j, normalized by the inter-feature distance between the matches (we use L_2). We compute the final updated weights $\{w_1,\ldots,w_{|X_i|}\}$ as the element-wise median of these $(C-1)$ vectors, normalized to maintain the original total weight. The final weights give a robust estimate of how much each feature consistently matched with other features in intra-cluster images. We normalize to maintain constant total weight per image, such that $\sum_{p=1}^{|X_i|} w_p = |X_i|$. This prevents weights attached to an irrelevant example from wasting away to nothing and getting stuck in their initial cluster.

Essentially, our method updates the feature weights of each image to produce tighter clusters in subsequent iterations. This is possible because the weight updates guarantee that the matching cost between two images decreases when compared to that obtained prior to the weight updates. Our method then chooses the weights (by the median) for each image such that the overall matching cost between intra-cluster images decreases.

In general, if a clump of images in a cluster contains instances of the same category, high weights will be attributed to their consistently re-occurring parts—the foreground. Note that depending on the current cluster membership, the updated feature weights can be quite different from the weights that were used to compute the clusters. To begin the next iteration, we re-compute the flows and affinities between all pairs of all N examples using the new weights, and re-cluster. As the weight distributions shift, subsequent least-cost matches are biased towards matching those features more likely to be foreground. We iterate between the matching, clustering, and re-weighting, until there is no change in the cluster assignments or until the average percent change in weight is below a threshold. In practice, we observe the most impact from the first several iterations (see Section 4).

Due to the complexity of computing the optimal matching on weighted point sets, in practice we compute the matrix $\mathbf{C}$ with a variant of the pyramid match kernel (PMK) algorithm [7]. The PMK approximates the least cost match for unweighted sets in linear time in the number of points in a set by intersecting multi-resolution histograms computed in the feature space (see [7]). Though defined for unweighted point sets, we can apply weights by scaling histogram counts by a point's weight upon insertion. Given two multi-resolution histograms, for every intersecting bin, we compute the optimal matching between the features from both sets that share the bin. We record the flow and cost that

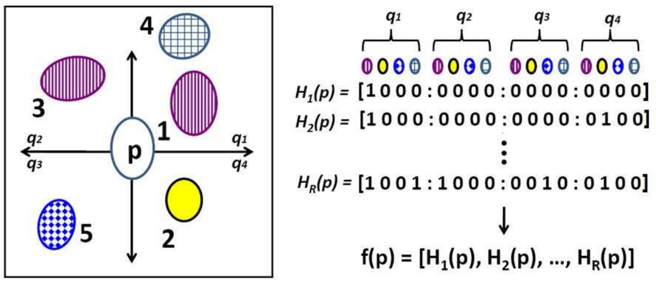

4
3
1
p
q2
q1
q3
q4
5
2
q1
q2
q3
q4
H1(p) = [1 0 0 0 : 0 0 0 0 : 0 0 0 0 : 0 0 0 0]
H2(p) = [1 0 0 0 : 0 0 0 0 : 0 0 0 0 : 0 1 0 0]
⋮
HR(p) = [1 0 0 1 : 1 0 0 0 : 0 0 1 0 : 0 1 0 0]
f(p) = [H1(p), H2(p), …, HR(p)]

Figure 2: Semi-local descriptor schematic for base feature p. Ellipses denote features, their patterns indicate the visual word types, numbers indicate the rank order of spatial proximity to the base feature, and q_i's denote quadrants relative to p. (Best viewed in color.)

each point at the current resolution level contributes to the match; any remaining weight is propagated to the next coarser pyramid level and can be used in future matchings. Zero-weighted features at any level do not contribute to the match. When all bins have been intersected, we have accumulated the approximate flow and match cost. Each per-bin flow computation is super-linear in the intersection value, but feature space partitions given by the pyramid result in small and gradually increasing intersection counts.

While the method in [8] uses weights to affect PMK matching cost, it does not compute flow fields that are influenced by the weights. More importantly, that clustering method is a one-shot process that does not benefit from the mutual updating between clusters and feature weights.

3.2 Semi-Local Proximity Distribution Descriptors

Our algorithm description thus far implies an orderless set-of-features representation. Local features are a favored representation due to their resilience under occlusion and clutter. Yet, too much locality can be problematic: features with minimal spatial extent may be too generic and easily matched to anything, and comparing unordered sets fails to enforce geometry. Thus, we propose a novel *semi-local* region descriptor that encodes the appearance and relative locations of other features in a spatial neighborhood. Our descriptor is inspired by the proximity distribution kernel [13], which compares images described by cumulative histograms of nearby visual word pairs. However, while their approach summarizes an entire image with one histogram, we design a proximity distribution feature for each interest point, which makes it possible to use rich local configuration cues within an explicit, weighted matching (and thus calculate the flow as described above).

We extract local patch features at all interest points. Then we construct a standard n-word visual vocabulary from a random pool of descriptors (we use SIFT [16]), and record each feature's word type. For each patch, for each of four directions (quadrants) relative to its center, we compute a cumulative distribution that counts the number of each type of visual word that occurs within that feature's r spatially nearest neighbor features, incremented over increasing values of r (see Figure 2).

More precisely, consider an image with patches $\{p_1,\dots,p_m\}$ and their associated word types $\{v_1,\dots,v_m\}$. For each p_i, we construct R total $4n$-dimensional histogram vectors $H_r(p_i)$, for $r = 1,\dots,R$. In each, the first n bins represent quadrant 1, the next n bins represent quadrant 2, and so on. Each n-length chunk is a histogram counting the number of occurrences of each word type v_j within p_i's r spatially nearest feature points, divided into quadrants relative to p_i. Note that higher values of r produce a vector $H_r(p_i)$ covering a spatially larger region. Finally, our semi-local descriptor for p_i is the concatenation of these R histograms: $f(p_i) = [H_1(p_i),\dots,H_R(p_i)]$.

Every patch's $R \times 4n$-length vector is a scale- and translation-invariant encoding of neighborhood appearance and coarse geometry. (We can add rotation invariance by setting quadrants based on a feature's dominant gradient; we have not yet explored this variant.) Due to the high-dimensionality and correlation among dimensions, we compute compact descriptors using PCA. Matching sets of our descriptors does not explicitly require spatial contiguity. Still, individual matches are dependent due to their spatial extent and overlap.

Discussion. What are the assumptions of our approach? For a pattern to be discovered, it must have support among multiple examples in the collection. Further, only visual patterns that share some configuration of similar semi-local regions can ever be found (e.g., our method will not discover a single cluster consisting of both soccer balls and volleyballs, but it can discover a group comprised of different people's faces). Finally, *some* support for a pattern must be detected in the initial iteration for progress towards refining that pattern to be made in the remaining iterations.

Note that features that are strictly speaking "background" can also earn high weights, if they happen to consistently re-occur with the same foreground class. So, what is learned depends on what the collection U contains: for example, if bikes are typically against a bike rack, then we can expect the pattern to be found as a single entity. The same holds for images with multiple objects that repeatedly co-occur—for example, if computer monitors always exist on desks. This is a natural outcome for unsupervised learning from static images (e.g., nothing can indicate that the bike and rack are not one composite object unless they often occur separately), and satisfies the problem definition.

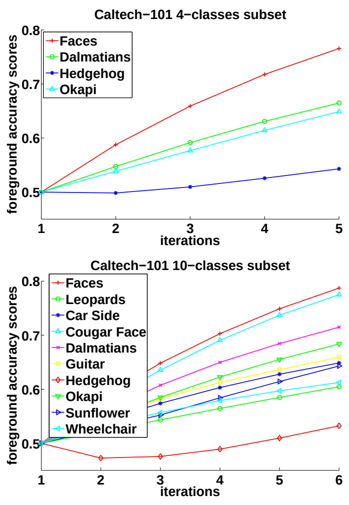
4 Results

We performed experiments both to analyze the mutual reinforcement of foreground and clusters, and to compare against existing unsupervised methods. We work with images from the Caltech-101, because the dataset provides object segmentations that we need as ground truth to evaluate our foreground detection. Unless otherwise specified below, we sample SIFT features at regular image intervals and discard any contrast-free regions.

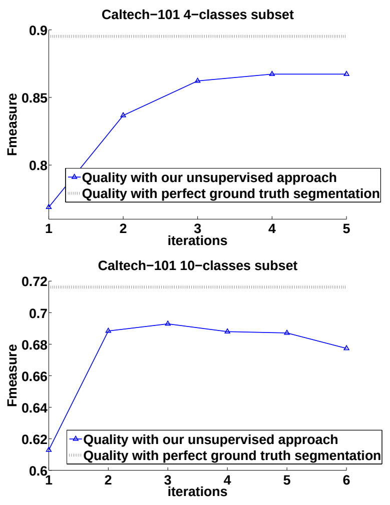
To determine when to stop iterating, we measure the percent change in the average feature weight change in all images from one iteration to the next, and stop once it slows to 15% or less (a threshold we set arbitrarily). When clustering, we set k as the number of classes present in the dataset in order to evaluate how well the true objects are discovered. We fix the neighborhood parameter at $R = 64$, following [13]. We set the vocabulary size n depending on the number of input images ($n = 200$ for the smaller sets, and $n = 400$ for the large ones) in an attempt to get good coverage.

Analyzing the Effects of Mutual Foreground/Clustering Reinforcement. While some classes in the Caltech-101 are fairly clutter-free, we purposely select classes with the highest clutter in order to demonstrate our method's impact. To do this, we first built *supervised* classifiers on all 101 categories: one trained with all features, and one trained using only foreground features. Then we ranked the classes for which segmentation most helped the supervised classifier, since these directed us to the classes with the most variable and confusing backgrounds. In this way, we formed a four-class (Faces, Dalmatians, Hedgehogs, and Okapi) and 10-class (previous four plus Leopards, Car_side, Cougar_face, Guitar, Sunflower, and Wheelchair) set. For each class, we use the first 50 images.

If our algorithm correctly identifies the important features, we expect those features to lie on the foreground objects since that is what primarily re-occurs in these datasets. To evaluate this, we compare the feature weights computed by our method with the ground



(a) Quality of foreground detection



(b) Quality of clusters formed

Figure 3: Evaluation of feature selection and category discovery. **(a)** The average foreground scores over iterations for all images from the 4-class (top) and 10-class (bottom) sets from the Caltech-101. **(b)** The cluster quality for those sets. The black dotted lines indicate the best possible quality that could be obtained if the ground truth segmentation were known (see text).

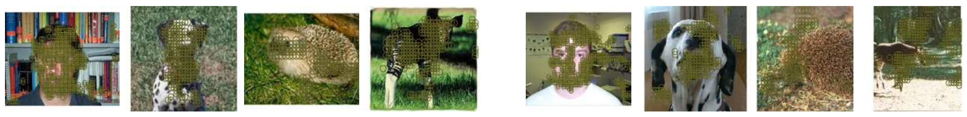


Figure 4: Examples showing the highest weighted features per image. In the left four examples, our method attributes weight only to foreground features. The right four are the examples our method does most poorly on: it weights foreground features highly, but also (mistakenly) finds good support for some background. (Best viewed in color.)

truth list of foreground features. We quantify accuracy by the percentage of total feature weight in an image that our method attributes to true foreground features. To make values comparable across images and classes, we compute $\frac{fg}{fg+bg}$, where fg and bg denote the sums of all foreground (background) weights normalized by the number of all foreground (background) features, respectively. If all weights were on foreground, the score would be 1, while if all weights were on background, the score would be 0.

Figure 3(a) shows our method's unsupervised foreground selection for the two datasets. All features start with uniform weights, which yields a base score of 0.5. Then each image's weights continually shift to the foreground, with significant gains for most classes as the clusters continue to be refined. In the 10-class set, the Hedgehog improves more slowly. Upon examination, we found that this was due to many hedgehogs dispersed across the initial clusters, resulting in more gradual convergence and cluster swaps.

As our method weights foreground features more highly, we also expect a positive effect on cluster quality. Since we know the true labels of each image, we can use the F-measure to measure cluster homogeneity. The F-measure measures the degree to which each cluster contains only and all objects of a particular class: $F = \sum_i \frac{n_i}{n} \max_j F'(i,j)$, where $F'(i,j) = \frac{2\times \mathrm{R}(i,j)\times \mathrm{P}(i,j)}{\mathrm{R}(i,j)+\mathrm{P}(i,j)}$, P and R denote precision and recall, respectively, and i indexes the classes and j indexes the clusters. High values indicate better quality. Fig-

	Affinity propagation [4]	Our method
Purity (7-classes) (%)	59.41	78.91
Purity (20-classes) (%)	36.91	65.61

	Partial match clusters [8]	Our method
Purity (%)	85.00 ± 4.72	88.82 ± 0.86
Prediction rate (%)	84.10 ± 5.07	87.13 ± 0.37

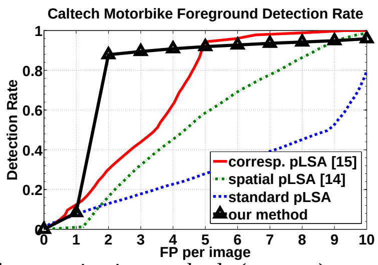


Figure 5: Comparison with existing unsupervised visual categorization methods (see text).

ure 3(b) shows the impact of foreground detection on cluster quality. To provide an upper bound on what quality level would result if we were to have *perfect* foreground segmentation, we also evaluate clusters obtained using *only* the foreground features (black dotted lines). Note that without any supervision or fg/bg annotation, our approach clusters almost as well as the ideal upper bound. Also, as we iterate, the better fg weights incrementally improve the clusters, until quality levels out. Figure 4 illustrates example results. Additional examples can be found in [11].

Comparison with Existing Unsupervised Methods. Next we empirically compare against results from alternative unsupervised visual learning methods [4, 8, 15, 14].

The authors of [4] propose a clustering algorithm called affinity propagation which considers all points as candidate exemplars and uses message passing to iteratively find the best set of exemplars that partition the data. They chose two subsets of the Caltech-101: a 20-class subset and a 7-class subset, taking the first 100 images of each class (see [4] for class names). In Figure 5 (top table) we compare our method with the same data, using the “purity” cluster quality measure used in [4]. Purity measures the extent to which a cluster contains images of a single dominant class, Purity $= \sum_j \frac{n_j}{n} \max_i \mathrm{P}(i, j)$. A strength of the affinity propagation method is that non-metric affinities are allowed, and so the authors compare images with SIFT features and a voting-based match, which is insensitive to clutter [16]. Still, the clusters found by our method are significantly more accurate, indicating the strength of both our refinement process and semi-local descriptor.

In Figure 5 (bottom table) we compare against the method of [8], which also forms groups with partial-match spectral clustering, but does not attempt to mutually improve foreground feature weights and clusters as our method does. Using the same 3,188 image Caltech-4 database and base features, our method gives better cluster purity as well as better prediction for novel examples. Results are averaged over 10 runs with randomly selected learning/testing pools. Our algorithm’s very small standard deviations in accuracy indicate that it is less sensitive to the composition of the unlabeled data, and provides significantly more reliable groupings.

Finally, the plot in Figure 5 compares the accuracy of our method’s foreground discovery to that of several latent topic models for the Caltech motorcycle class, as reported in [15]. The fg detection rate is computed by varying the threshold among the top 20% most confident features as prescribed in [15]. We compare our method with a standard pLSA model, pLSA with spatial information [14], and a correspondence-based pLSA variant [15]. The pLSA models compute foreground confidence based on the probability of the topic given the patch. Our approach outperforms the others for most points on the detection curve, providing much better precision for low false positive rates.

Conclusions and Future Work. We have introduced a novel unsupervised method for discovering foreground features in images. Clusters are determined by matching weighted feature sets, and weights are iteratively adjusted based on contributions to intra-cluster image matches. We show that this mutual reinforcement improves both cluster quality and foreground detection, with datasets containing four to twenty categories.

In future work, we will investigate how our algorithm could accept incremental updates to the unlabeled pool, and extend it to multiple-label cluster assignments. We also plan to directly evaluate our semi-local region descriptor against related alternatives.

Acknowledgements. This research is supported in part by NSF CAREER award 0747356, Microsoft Research, and the Henry Luce Foundation. We would like to thank David Liu and Delbert Dueck for sharing their experimental data.

References

[1] A. Agarwal and B. Triggs. Hyperfeatures Multilevel Local Coding for Visual Recognition. In *ECCV*, 2006.

[2] O. Chum and A. Zisserman. An Exemplar Model for Learning Object Classes. In *CVPR*, 2007.

[3] G. Dorko and C. Schmid. Selection of Scale-Invariant Parts for Object Class Recognition. In *ICCV*, 2003.

[4] D. Dueck and B. Frey. Non-metric Affinity Propagation for Unsupervised Image Categorization. In *ICCV*, 2007.

[5] J. Dy and C. Brodley. Feature Selection for Unsupervised Learning. *JMLR*, 5:845–889, 2004.

[6] R. Fergus, L. Fei-Fei, P. Perona, and A. Zisserman. Learning Object Categories from Google's Image Search. In *ICCV*, 2005.

[7] K. Grauman and T. Darrell. The Pyramid Match Kernel: Discriminative Classification with Sets of Image Features. In *ICCV*, 2005.

[8] K. Grauman and T. Darrell. Unsupervised Learning of Categories from Sets of Partially Matching Image Features. In *CVPR*, 2006.

[9] S. Lazebnik, C. Schmid, and J. Ponce. A Sparse Texture Representation Using Affine-Invariant Regions. In *CVPR*, 2003.

[10] S. Lazebnik, C. Schmid, and J. Ponce. Semi-Local Affine Parts for Object Recognition. In *BMVC*, 2004.

[11] Y. J. Lee and K. Grauman. Foreground Focus: Unsupervised Learning From Partially Matching Images. Technical report, UT-Austin, May 2008.

[12] B. Leibe, A. Leonardis, and B. Schiele. Combined Object Categorization and Segmentation with an Implicit Shape Model. In *Wkshp on Statistical Learning in Computer Vision*, 2004.

[13] H. Ling and S. Soatto. Proximity Distribution Kernel for Geometric Context in Recognition. In *ICCV*, 2007.

[14] D. Liu and T. Chen. Semantic-Shift for Unsupervised Object Detection. In *CVPR Wkshop on Beyond Patches*, 2006.

[15] D. Liu and T. Chen. Unsupervised Image Categorization and Object Localization using Topic Models and Correspondences between Images. In *ICCV*, 2007.

[16] D. Lowe. Distinctive Image Features from Scale-Invariant Keypoints. *IJCV*, 60(2), 2004.

[17] M. Marszalek and C. Schmid. Spatial Weighting for Bag-of-Features. In *CVPR*, 2006.

[18] T. Quack, V. Ferrari, B. Leibe, and L. Van Gool. Efficient Mining of Frequent and Distinctive Feature Configurations. In *ICCV*, 2007.

[19] P. Quelhas, F. Monay, J.-M. Odobez, D. Gatica-Perez, T. Tuytelaars, and L. Van Gool. Modeling Scenes with Local Descriptors and Latent Aspects. In *ICCV*, 2005.

[20] C. Rother, V. Kolmogorov, T. Minka, and A. Blake. Cosegmentation of Image Pairs by Histogram Matching - Incorporating a Global Constraint into MRFs. In *CVPR*, 2006.

[21] Y. Rubner, C. Tomasi, and L. Guibas. The Earth Mover's Distance as a Metric for Image Retrieval. *IJCV*, 40(2):99–121, 2000.

[22] B. Russell, A. Efros, J. Sivic, W. Freeman, and A. Zisserman. Using Multiple Segmentations to Discover Objects and their Extent in Image Collections. In *CVPR*, 2006.

[23] J. Shi and J. Malik. Normalized Cuts and Image Segmentation. *TPAMI*, 22(8):888–905, August 2000.

[24] J. Sivic and A. Zisserman. Video Data Mining Using Configurations of Viewpoint Invariant Regions. In *CVPR*, 2004.

[25] M. Weber, M. Welling, and P. Perona. Unsupervised Learning of Models for Recognition. In *ECCV*, 2000.

[26] J. Winn and N. Jojic. LOCUS: Learning Object Classes with Unsupervised Segmentation. In *ICCV*, 2005.