Int J Comput Vis (2014) 108:115–132
DOI 10.1007/s11263-013-0679-z

Iterative Category Discovery via Multiple Kernel Metric Learning

Carolina Galleguillos · Brian McFee · Gert R. G. Lanckriet

Received: 26 February 2013 / Accepted: 21 November 2013 / Published online: 7 December 2013

Abstract The goal of an object category discovery system is to annotate a pool of unlabeled image data, where the set of labels is initially unknown to the system, and must therefore be discovered over time by querying a human annotator. The annotated data is then used to train object detectors in a standard supervised learning setting, possibly in conjunction with category discovery itself. Category discovery systems can be evaluated in terms of both accuracy of the resulting object detectors, and the efficiency with which they discover categories and annotate the training data. To improve the accuracy and efficiency of category discovery, we propose an iterative framework which alternates between optimizing nearest neighbor classification for known categories with multiple kernel metric learning, and detecting clusters of unlabeled image regions likely to belong to a novel, unknown categories. Experimental results on the MSRC and PASCAL VOC2007 data sets show that the proposed method improves clustering for category discovery, and efficiently annotates image regions belonging to the discovered classes.

Keywords Category discovery · Metric learning · Multiple kernel learning · Iterative discovery

C. Galleguillos (✉)
SET Media Inc, San Francisco, CA, USA
e-mail: cgalleguillos@gmail.com

B. McFee
Columbia University, New York, NY, USA
e-mail: brm2132@columbia.edu

G. R. G. Lanckriet
University of California, San Diego, La Jolla, CA, USA
e-mail: gert@ece.ucsd.edu

1 Introduction

The acquisition of large collections of accurately labeled training examples for learning object models has become one of the most important problems in the field of object recognition. When gathering labeled examples, the limiting factor is most often the amount of manual labor required to produce and curate the labels. The bottleneck of human annotator effort has inspired researchers to develop more efficient schemes to collect high-quality, labeled training data for supervised object recognition algorithms (Collins et al. 2008; Vijayanarasimhan and Grauman 2009; Branson et al. 2010; Tian et al. 2007).

In general, the set of potential object labels is effectively unbounded, and may grow over time. Rather than requiring annotations to be drawn from a small, fixed vocabulary, we may prefer a system which can continually and automatically *discover* object categories. Category discovery typically occurs in two phases (which may be repeated sequentially Lee and Grauman 2010): (1) group unlabeled image regions under some appropriate notion of similarity, and (2) request labels for the resulting groups from a human annotator. These two steps highlight the design goals of a category discovery framework. First, the system must discover semantically coherent sets of image regions, a task which will ultimately depend upon the quality of the similarity metric. Second, the system should be both accurate and efficient in its requirements on the human annotator.

In this work, we propose a novel iterative discovery framework that uses multiple kernel metric learning to improve category discovery. Our framework, as shown in Fig. 1, uses an initial set of familiar categories to learn a distance function over image regions. Using the optimized distance, all unfamiliar regions are grouped into clusters, and the most prominent cluster is retrieved, and interactively annotated by

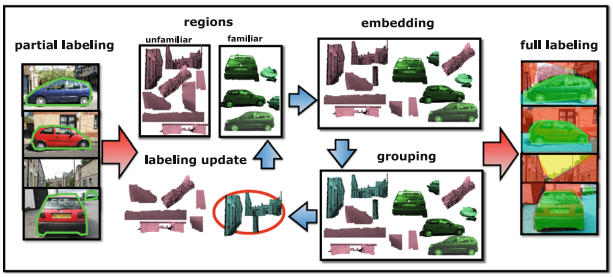

Fig. 1 A set of images is partially labeled with familiar categories (e.g., car), while the remaining regions are left unlabeled. Both labeled (familiar) and unlabeled regions are used to optimize a similarity space. At each iteration, a new object category is discovered, and regions belonging to the new category are now used to update the similarity space. The system terminates when all regions have been labeled

a human as a potentially new category. Unlike previous work, our interactive annotation model maintains a correctly annotated training set, but still reduces the total effort required by the human annotator. Finally, after each new set of labeled regions is discovered, the distance function is updated to optimize nearest neighbor retrieval, and the process repeats until all instances are labeled.

1.1 Related Work

Previous work on category discovery has generally focused on learning all categories at once via unsupervised learning (Tuytelaars et al. 2010; Lee and Grauman 2010), rather than the iterative (one-at-a-time) approach taken here.

Unsupervised learning methods can be effective for category discovery, as they require no labeled training data and merely seek to discover latent structure in the data, e.g., clusters (Grauman and Darrell 2006; Russell et al. 2006; Sivic et al. 2005; Todorovic and Ahuja 2006; Zhu et al. 2012; Faktor and Irani 2012) or hierarchies (Bart et al. 2008; Sivic et al. 2008) rather than learn explicit object models. The general strategy of unsupervised methods is to uncover groupings of images (or image regions) that share visual patterns, with the hope that the majority of the images within a group come from the same object category.

As region groupings are commonly found via clustering or topic modeling (Russell et al. 2006; Lee and Grauman 2010; Galleguillos et al. 2011), the quality of any unsupervised category discovery system will ultimately depend upon how it determines similarity between image regions, and how it finds groupings that result in new categories. Region similarity may be defined in any number of ways, e.g., deriving from feature descriptors, contextual cues, and so on. Recent work has examined algorithms which optimize similarity for classification (given labeled examples) (Frome et al. 2007; Wang et al. 2010) or derive similarity from classifier response on a set of known categories (Kang et al. 2012). However, to the best of our knowledge, there has thus far been no study of systematically optimizing a similarity function for use in iterative category discovery.

The work of Lee and Grauman (2011) is the closest to our work as it treats category discovery as an iterative procedure. The method focuses on selecting specific object instances instead of optimizing the similarity space where all instances are represented. The method approaches the "easier" instances first and then expands towards more complex instances by starting with a fixed number of *stuff* (objects of amorphous spatial extent) classifiers to discover classes of *things* (rigid objects) (Forsyth et al. 1995; Heitz and Koller 2008), one category at a time. After each iteration, a new discriminative object detector is trained and added to the system, and the process repeats. In contrast with aggregating classifiers and combining their outputs in each iteration, we start the discovery process with few arbitrary object classes and uncover new categories one class at a time and update the distance metric accordingly. As a result, the proposed method retains a fixed number of parameters to define the distance metric, rather than increasing the parameter space by adding new discriminative classifiers (e.g., support vector machines) at each iteration. Table 1 shows the key differences between both strategies.

To our knowledge there are no other methods that learn a similarity metric iteratively for object category discovery. There is extensive work on learning a single similarity metric from multiple kernels for multi-class problems (Gehler and Nowozin 2009; Varma and Ray 2007; Vedaldi et al. 2009).

Table 1 Comparison between this work and Lee and Grauman (2011)

Method	Objective	Input	Labeling	Iterative approach
LG11	Clustering and segmentation accuracy	*Stuff* classes	Majority vote	Identify easy instances first and aggregate binary classifiers at each iteration
Our framework	Efficiency and label accuracy	Arbitrary classes	Largest subset	Optimize similarity metric at every iteration

The "objective" column corresponds to the specific focus of each algorithm, "input" is the type of classes required to perform discovery, "labeling" is the strategy taken to label the segments as a new class, and "iterative approach" corresponds to how the framework learns a new model and incorporates the newly labeled segments

Although using a different similarity metric for each class has been shown to perform well on these tasks, it is difficult to scale to large datasets with many classes, and the predictions from each classifier must be combined to yield a single prediction. Galleguillos et al. (2010) describe a system to learn a unified similarity metric that integrates appearance with contextual features for detecting multiple object categories. Their multiple kernel learning algorithm is the closest to our work, as it learns a unified similarity metric for nearest neighbor classification.

1.2 The Proposed Framework

Initially, our framework assumes that a set of training images has been partially annotated with a set of known (*familiar*) categories, so that image regions corresponding to familiar categories have been labeled (Fig. 1). The initial familiar categories may be chosen arbitrarily, and we do not place any assumptions on which types of objects are initially labeled (e.g., "objectness"), their contextual interactions or use classifiers trained with external data, as is done by other frameworks (Lee and Grauman 2011). All remaining, initially unlabeled image regions are assumed to belong to *unfamiliar categories*.

Category discovery proceeds by clustering the unlabeled image regions, and then querying for labels within the most prominent cluster. Regions belonging to the majority category within the cluster are labeled, and all remaining regions are kept unlabeled for the next round (Fig. 2). Efficient category discovery therefore relies upon the quality of the clustering (i.e., label purity and size of clusters), which in turn relies upon the quality of the underlying notion of region similarity, i.e., the distance metric over region descriptors.

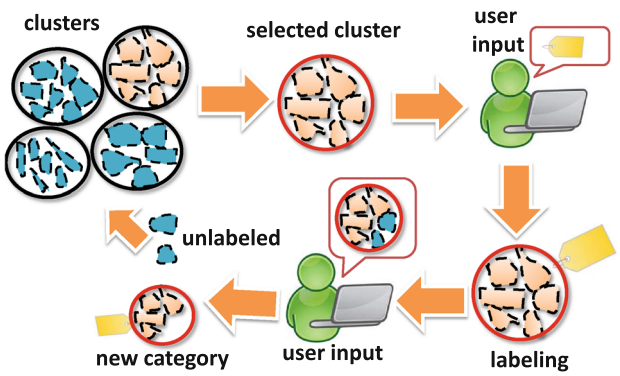

Fig. 2 Cluster labeling: the most prominent cluster is selected and presented to a human annotator. The annotator selects the majority category label of the objects in the cluster, and either (a) selects instances that correspond to the majority category, or (b) removing instances that don't belong to the majority category. Regions belonging to the largest subset within the cluster are labeled, and all remaining regions are kept unlabeled for the next round

If we were provided with a fully labeled training set, the distance metric could be optimized in one step to separate all categories via standard techniques (Weinberger et al. 2006; Galleguillos et al. 2010). However, because we do not know to *which* category an unlabeled image region may belong, we cannot directly optimize a similarity function to discriminate between unfamiliar categories. Instead, we start by training a similarity metric to discriminate between familiar categories using k-nearest-neighbor retrieval. Our decision to optimize for nearest neighbor accuracy is motivated by two ideas: first, improving nearest neighbor provides a direct way to group together unfamiliar regions (clustering under the learned metric), and second, nearest neighbor naturally supports arbitrary numbers of classes.

Moreover, because nearest-neighbor classification is inherently multi-class, it automatically extends to *new* classes without expanding the set of parameters to be learned. We see this as a key advantage over previous methods, where the detection of unfamiliar categories derives from the output of a growing collection binary classifiers trained on specific familiar categories (Lee and Grauman 2010, 2011).

1.3 Our Contributions

Our main technical contribution is an iterative category discovery framework, based on a multiple-kernel extension to the metric learning to rank (MLR) algorithm, which learns an optimized distance metric over multiple, heterogeneous input features. Additionally, we propose an interactive annotation model which balances the efficiency of majority-vote cluster labeling (Lee and Grauman 2011) with the accuracy of one-at-a-time labeling.

The proposed method discovers object categories by grouping image regions under the (iteratively) optimized distance metric. We present results of the method on diverse datasets, and demonstrate that optimizing the distance metric can significantly improve both efficiency and accuracy compared to the native, unoptimized distance.

2 Iterative Category Discovery

We will first introduce notation and formalize the problem. Table 2 gives a brief summary of the notation used throughout this article.

Each image $\mathcal{I}$ in the image set is partitioned into segments x_i. We assume that each segment x_i belongs to exactly one object of class ℓ_i.

A labeled region is *familiar*, meaning that the label belongs to the set of currently known categories. An unlabeled region can be either *familiar* or *unfamiliar*, depending on whether its true label belongs to the set of currently known categories. Familiar and unlabeled image regions are

Table 2 Notation used throughout this article.

Symbols	Definition
$\mathcal{I}$	Image
x_i	Image segment (region)
$\mathcal{L}$	Set of known object classes, familiar labels
ℓ_0	unknown object class, unfamiliar label
$\mathcal{X}_\mathrm{m} = \{x_1, x_2, \dots\}$	Labeled training regions (ground truth segmentation)
$\mathcal{X}_\mathrm{f}$	Labeled training segments (automatic segmentation)
$\mathcal{X}_\mathrm{u}$	Unlabeled segments (automatic segmentation)
$\phi_t(\cdot)$	Feature map for kernel t
$W \succeq 0$	Positive semi-definite matrix
$\|x - y\|_W$	Mahalanobis distance defined by W

collected into the sets $\mathcal{X}_\mathrm{f}$ and $\mathcal{X}_\mathrm{u}$ respectively. All segments are then collected to form the region set $\mathcal{X} = \mathcal{X}_\mathrm{f} \cup \mathcal{X}_\mathrm{u}$, which is used to label all regions in $\mathcal{X}_\mathrm{u}$ via iterative category discovery (see Algorithm 1).

Each image segment $x_i \in \mathcal{X}$ is represented by a collection of m features $\phi_t(x_i)$ $(t \in 1, 2, \dots, m)$, where each $\phi_t(x_i)$ represents x_i in a reproducing kernel Hilbert space characterized by a corresponding kernel function

$$k_t(x_i, x_j) = \left\langle \phi_t(x_i), \phi_t(x_j) \right\rangle.$$

At each iteration, object category discovery proceeds as follows (illustrated in Fig. 3):

1. A unified similarity metric is learned on a collection of n labeled points in the m feature spaces.
2. Unlabeled regions $\mathcal{X}_u$ are grouped by hierarchical clustering with the optimized similarity metric.
3. The cluster c^* with the smallest average intra-cluster distance is selected, and queried for the (possibly novel) majority within-cluster label ℓ^*.
4. All regions $x_i \in c^*$ belonging to ℓ^* are selected and added to $\mathcal{X}_\mathrm{f}$ with label ℓ^*, and the remaining regions are kept unlabeled.
5. The metric is updated by training again with the additional, newly labeled regions. The entire process repeats until there are no remaining unlabeled regions.

As the learned space is optimized at each iteration for nearest neighbor retrieval, we expect that similar points in the space (belonging to the same class) are drawn closer, thus generating clusters that are larger, tighter, and purer. Consequently, iterative category discovery with the learned metric should require fewer iterations to label all regions (compared to using the native metric), and generalize better on novel test regions. When clusters are tight, a visually homogenous cluster is likely to be selected by the algorithm. If the selected cluster is large, many segments may be labeled in each iteration, and if the cluster also has high purity, then the majority of the segments within the cluster will be labeled.

3 Optimizing the Space

The first step of our framework consists of learning an optimized similarity function over image regions. Note that we cannot know *a priori* which features will be discriminative for unfamiliar categories. We therefore opt to include many different descriptors, capturing texture, color, scene-level context, etc. (see Sect. 5.1.) In order to effectively integrate heterogeneous features, we turn to multiple kernel learning (MKL) (Lanckriet et al. 2004). While MKL algorithms have been widely applied in computer vision applications (Varma and Ray 2007; Vedaldi et al. 2009), most research has focused on binary classifiers (i.e., support vector machines), with rel-

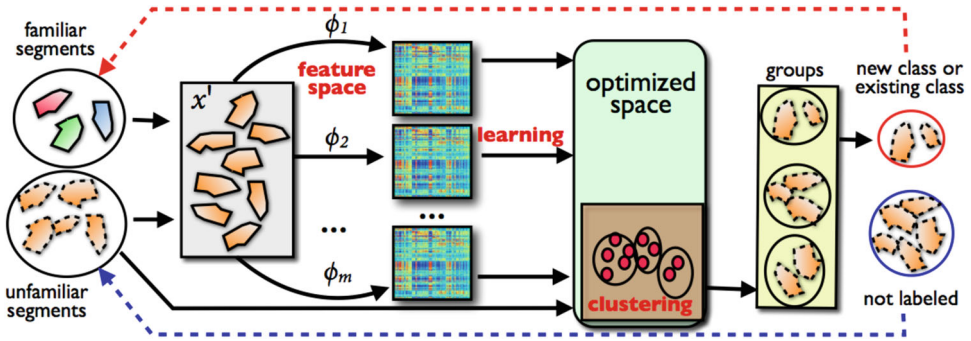

Fig. 3 Iterative object class discovery: Initially, images are partitioned into multiple segments, each of which are mapped into multiple feature spaces, and then projected into the optimized similarity space learned by MKMLR (Algorithm 3). Unlabeled segments are clustered in the optimized space, and the cluster with the minimum intra-class average distance is selected as a new class. A human annotator then supplies a (possibly new) label for the largest category of segments within the cluster. The newly labeled regions are addressed now as "familiar" segments and used to retrain the distance metric. The process is repeated until all regions are labeled.

Algorithm 1 Iterative category discovery

Input: Labeled segments $\mathcal{X}_\mathrm{f}$, unlabeled segments $\mathcal{X}_\mathrm{u}$
for each $x_i \in \mathcal{X}_\mathrm{f} \cup \mathcal{X}_\mathrm{u}$ **do**
  extract features $\phi_t(x_i)$ $(t \in 1, 2, \ldots, m)$
end for
repeat
  learn the distance metric W from segments $\mathcal{X}_\mathrm{f} \cup \mathcal{X}_\mathrm{u}$ (Algorithm 3)
  cluster $\mathcal{X}_\mathrm{u}$ under $W \rightarrow$ clusters $\{c_1, c_2, \cdots, c_k\}$
  $c^* \leftarrow \operatorname{argmin} d_\mathrm{intra}(c_j)$ (See Eq. 14)
  query the majority category ℓ^* within c^*
  $c_+ \leftarrow \{x_i : x_i \in c^* \wedge \ell(x_i) = \ell^*\}$
  $c_- \leftarrow \{x_i : x_i \in c^* \wedge \ell(x_i) \neq \ell^*\}$
  $\mathcal{X}_\mathrm{f} \leftarrow \mathcal{X}_\mathrm{f} \cup c_+$
  $\mathcal{X}_\mathrm{u} \leftarrow \mathcal{X}_\mathrm{u} \setminus c_+$
  $\mathcal{L} \leftarrow \mathcal{L} \cup \ell^*$
until all x_i are labeled

atively little attention given to the optimization of nearest neighbor classification and retrieval.

Recently, multiple kernel large margin nearest neighbor (MKLMNN) has been proposed as a method for integrating heterogeneous data in a nearest-neighbor setting (Galleguillos et al. 2010). Like the original LMNN algorithm (Weinberger et al. 2006), MKLMNN attempts to find a linear projection of data such that each point's target neighbors (i.e., those with similar labels) are drawn closer than dissimilar neighbors by a large margin. While this notion of distance margins is closely related to nearest neighbor prediction, it does not optimize for the actual nearest neighbor accuracy.

Instead, we will derive a multiple kernel extension of the metric learning to rank algorithm (MLR) (McFee and Lanckriet 2010), which optimizes nearest neighbor retrieval more directly by examining the ordering of points generated by the learned metric. Before deriving the multiple kernel extension, we first briefly review the MLR algorithm for the linear case.

3.1 Metric Learning to Rank

Metric learning to rank (MLR, Algorithm 2) (McFee and Lanckriet 2010) is a metric learning extension of the Structural SVM algorithm for optimizing ranking losses (Joachims 2005; Tsochantaridis et al. 2005).

Whereas SVM$^\mathrm{struct}$ learns a vector $w \in \mathbb{R}^d$, MLR learns a positive semi-definite matrix W (denoted $W \succeq 0$) which defines a distance

$$d_W(i, j) := \|i - j\|_W^2 = (i - j)^\mathsf{T} W (i - j).$$

MLR optimizes W by evaluating the quality of rankings generated by ordering the training data by increasing distance from a query point. Ranking quality may be evaluated and optimized according to any of several metrics, including precision-at-k, area under the ROC curve, mean average precision (MAP), etc. Note that k-nearest neighbor accuracy can also be interpreted as a performance measure over rankings induced by distance.

Although ranking losses are discontinuous and non-differentiable functions over permutations, SVM$^\mathrm{struct}$ and MLR resolve this issue by encoding constraints for each training point as listed in Algorithm 2. Here, $\mathcal{X}$ is the training set of n points, $\mathcal{Y}$ is the set of all possible rankings (i.e., permutations of $\mathcal{X}$), y_x is the true or *best* ranking[1] for $x \in \mathcal{X}$, $\Delta(y_x, y)$ is the loss incurred for predicting y instead of y_x (e.g., decrease in precision-at-k), and ξ_x is a slack variable as in the standard soft-margin SVM (Cortes and Vapnik 1995). $\langle W, \psi(x, y)\rangle_\mathsf{F}$ is the *score* function which evaluates how well the metric W agrees with the input–output pair (x, y), encoded by the feature map ψ.

Algorithm 2 Metric Learning to Rank (McFee and Lanckriet, 2010)

Input: data $\mathcal{X} = \{x_1, x_2, \ldots, x_n\} \subset \mathbb{R}^d$,
  true rankings $y_1, y_2, \ldots y_n$,
  slack trade-off $C > 0$
Output: $d \times d$ matrix $W \succeq 0$

$$\min_{W \succeq 0,\ \xi} \quad \mathrm{tr}(W) + \frac{C}{n} \sum_{x \in \mathcal{X}} \xi_x$$

$$\text{s.t.}\ \forall x \in \mathcal{X}, \quad \forall y \in \mathcal{Y}:$$

$$\langle W, \psi(x, y_x)\rangle_\mathsf{F} \geq \langle W, \psi(x, y)\rangle_\mathsf{F} + \Delta(y_x, y) - \xi_x$$

To encode input–output pairs, MLR uses a variant of the *partial order feature* (Joachims 2005) adapted for distance ranking:

$$\psi(x, y) := \sum_{i \in \mathcal{X}_x^+,\ j \in \mathcal{X}_x^-} y_{ij} \frac{D(x, i) - D(x, j)}{|\mathcal{X}_x^+| \cdot |\mathcal{X}_x^-|}$$

$$D(x, i) := -(x - i)(x - i)^\mathsf{T}. \tag{1}$$

Here, $\mathcal{X}_x^+$ and $\mathcal{X}_x^- \subseteq \mathcal{X}$ denote the sets of positive and negative results with respect to example x (i.e., points of the *same category* or *different category*), and

$$y_{ij} := \begin{cases} +1 & \text{if } i \text{ precedes } j \text{ in } y \\ -1 & \text{if } j \text{ precedes } i \text{ in } y \end{cases}.$$

With this choice of ψ, the rule to predict y for a test point x is to simply sort $i \in \mathcal{X}$ in descending order of

$$\langle W, D(x, i)\rangle_\mathsf{F} = -\langle W, (x - i)(x - i)^\mathsf{T}\rangle_\mathsf{F} = -\|x - i\|_W^2. \tag{2}$$

Equivalently, sorting by increasing distance $\|x - i\|_W$ yields the ranking needed for nearest neighbor retrieval.

[1] In this setting, a *true ranking* is any ranking which places all relevant results before all irrelevant results.

Note that unlike classification-based metric learning algorithms (e.g., Weinberger et al. 2006), MLR readily supports the inclusion of unlabeled data. For a labeled point x_i, the irrelevant set $\mathcal{X}_i^-$ may include unlabeled (unfamiliar) examples $\mathcal{X}_u$, and no additional label structure or constraints need be assumed between the unlabeled examples. In the context of iterative category discovery, this enables the algorithm to push unfamiliar regions away from labeled (familiar) regions, without having to collapse unlabeled data together (by assuming a single *no label* category for all unlabeled regions) or pushing all unlabeled data apart (by assuming an individual dummy category for each unlabeled region).

Although Algorithm 2 lists exponentially many constraints, cutting-plane techniques can be applied to quickly find an approximate solution (Joachims et al. 2009).

3.2 Multiple Kernel Metric Learning

The MLR algorithm, as described in the previous section, produces a linear transformation of vectors in $\mathbb{R}^d$. In this section, we first extend the algorithm to support non-linear transformations via kernel functions, and then to jointly learn transformations of multiple kernel spaces.

3.2.1 Kernel MLR

Typically, non-linear variants of structural SVM algorithms are derived by observing that the SVM$^{\text{struct}}$ dual program can be expressed in terms of the inner products (or kernel function) between feature maps: $\langle \psi(x_1, y_1), \psi(x_2, y_2) \rangle$. (see, Tsochantaridis et al. 2005.) However, to preserve the semantics of distance ranking (Eq. 2), it would be more natural to apply non-linear transformations directly to x while preserving linearity in the structure $\psi(x, y)$. We therefore take an alternative approach in deriving kernel MLR, which is more in line with previous work in non-linear metric learning (Globerson and Roweis 2007; Galleguillos et al. 2010).

We first note that by combining Eqs. 1 and 2 and exploiting linearity of ψ, the score function can be expressed in terms of learned distances:

$$\begin{aligned} S(W, x, y) &:= \langle W, \psi(x, y) \rangle_{\mathsf{F}} \\ &= \sum_{i \in \mathcal{X}_x^+, j \in \mathcal{X}_x^-} y_{ij} \frac{\|x - j\|_W^2 - \|x - i\|_W^2}{|\mathcal{X}_x^+| \cdot |\mathcal{X}_x^-|}. \end{aligned} \tag{3}$$

Let $\phi : \mathcal{X} \to \mathcal{H}$ denote a feature map from $\mathcal{X}$ to a reproducing kernel Hilbert space (RKHS) $\mathcal{H}$. Inner products in $\mathcal{H}$ are computed by a kernel function

$$k(x_1, x_2) = \langle \phi(x_1), \phi(x_2) \rangle_{\mathcal{H}}.$$

Let $L : \mathcal{H} \to \mathbb{R}^n$ be a linear operator on $\mathcal{H}$ which will define our learned metric, and let $\|L\|_{\text{HS}}$ denote the Hilbert-Schmidt operator norm [2] of L.

Next, we define a score function in terms of L, which, as in Eq. 3, compares learned distances:

$$\begin{aligned} S_{\mathcal{H}}(L, x, y) &:= \sum_{i \in \mathcal{X}_x^+, j \in \mathcal{X}_x^-} y_{ij} \frac{d_L(x, j) - d_L(x, i)}{|\mathcal{X}_x^+| \cdot |\mathcal{X}_x^-|}. \\ d_L(x, i) &:= \|L(\phi(x)) - L(\phi(i))\|^2 \end{aligned} \tag{4}$$

We may now formulate an optimization problem similar to Algorithm 2 in terms of L:

$$\begin{aligned} L^* = \operatorname*{argmin}_{L, \xi} \ & \|L\|_{\text{HS}}^2 + \frac{C}{n} \sum_{x \in \mathcal{X}} \xi_x \qquad \text{s.t.} \\ \forall x, y : \ & S_{\mathcal{H}}(L, x, y_x) \geq S_{\mathcal{H}}(L, x, y) + \Delta(y_x, y) - \xi_x. \end{aligned} \tag{5}$$

The choice of $\|L\|_{\text{HS}}^2$ as a regularizer on L allows us to invoke the generalized representer theorem (Schölkopf et al. 2001). It follows that an optimum L^* of Eq. 5 admits a representation of the form

$$L^* = M \Phi^{\mathsf{T}},$$

where $M \in \mathbb{R}^{n \times n}$, and $\Phi \in \mathcal{H}^n$ contains the training set in feature space: $\Phi_x = \phi(x)$. By defining $W = M^{\mathsf{T}} M$ and $K = \Phi^{\mathsf{T}} \Phi$, we observe two facts:

$$\begin{aligned} d_L(x, i) &= \|L^* \phi(x) - L^* \phi(i)\|^2 \\ &= \|M \Phi^{\mathsf{T}} \phi(x) - M \Phi^{\mathsf{T}} \phi(i)\|^2 \\ &= \|K_x - K_i\|_{M^{\mathsf{T}} M}^2 \\ &= \|K_x - K_i\|_W^2, \end{aligned} \tag{6}$$

$$\text{and} \quad \|L^*\|_{\text{HS}}^2 = \operatorname{tr}(WK), \tag{7}$$

where for any z, $K_z := \Phi^{\mathsf{T}} \phi(z) = [k(x, z)]_{x \in \mathcal{X}}$ is a column vector of the kernel function evaluated at a point z and all training points x.

Note that the constraints in Eq. 5 render the program non-convex in L, which may itself be infinite-dimensional and therefore impossible to optimize directly. However, by substituting Eq. 6 into Eq. 4, we recover a score function of the same form as Eq. 3, except with x, i and j replaced by their corresponding kernel vectors K_x, K_i and K_j. We may then define the kernelized metric partial order feature:

[2] The Hilbert-Schmidt norm is a natural generalization of the Frobenius norm. For our purposes, this can be understood as treating L as a collection of n elements $v_i \in \mathcal{H}$ (one per output dimension of L), and summing over the squared-norms: $\|L\|_{\text{HS}} = \sqrt{\sum_i \langle v_i, v_i \rangle_{\mathcal{H}}}$.

$$\psi^K(x, y) := \sum_{i \in \mathcal{X}_x^+,\ j \in \mathcal{X}_x^-} y_{ij} \frac{D^K(x, i) - D^K(x, j)}{|\mathcal{X}_x^+| \cdot |\mathcal{X}_x^-|}$$

$$D^K(x, i) := -(K_x - K_i)(K_x - K_i)^\mathsf{T}. \tag{8}$$

Thus, at an optimum L^*, the score function can be represented equivalently as

$$S_\mathcal{H}(L^*, x, y) = \langle W, \psi^K(x, y) \rangle_\mathsf{F}. \tag{9}$$

Taken together, Eqs. 7 and 9 allow us to re-formulate Eq. 5 in terms of W and K, and obtain a convex program similar to Algorithm 2. The resulting program may be seen as a special case of Algorithm 3.

3.2.2 Multiple Kernel MLR

To extend the above derivation to the multiple kernel setting, we must first define how the kernels will be combined. Let $\mathcal{H}_1, \mathcal{H}_2, \ldots, \mathcal{H}_m$ each denote an RKHS, each equipped with corresponding kernel functions $k_1, k_2, \ldots, k_m$ and feature maps $\phi_1, \phi_2, \ldots, \phi_m$. From each space $\mathcal{H}_t$, we will learn a corresponding linear projection L_t. Each L_t will project to a subspace of the output space, so that each point x is embedded according to

$$x \mapsto \{\phi_t(x)\}_{t=1}^m \mapsto [L_t(\phi_t(x))]_{t=1}^m \in \mathbb{R}^{nm},$$

where $[\cdot]_{t=1}^m$ denotes the concatenation of projections $L_t(\phi_t(x))$. The (squared) Euclidean distance between the projections of two points x and j is

$$d_\mathrm{M}(x, j) = \sum_{t=1}^m \|L_t(\phi_t(x)) - L_t(\phi_t(j))\|^2. \tag{10}$$

If we substitute Eq. 10 in place of d_L in Eq. 4, we can define a multiple-kernel score function S_M. By linearity, this can be decomposed into the sum of single-kernel score functions:

$$\begin{aligned} S_\mathrm{M}(\{L_t\}, x, y) &:= \sum_{i \in \mathcal{X}_x^+,\ j \in \mathcal{X}_x^-} y_{ij} \frac{d_\mathrm{M}(x, j) - d_\mathrm{M}(x, i)}{|\mathcal{X}_x^+| \cdot |\mathcal{X}_x^-|} \\ &= \sum_{t=1}^m S_{\mathcal{H}_t}(L_t, x, y). \end{aligned} \tag{11}$$

Again, we formulate an optimization problem as in Eq. 5 by regularizing each L_t independently:

$$\min_{\{L_t\}, \xi} \sum_{t=1}^m \|L_t\|_\mathrm{HS}^2 + \frac{C}{n} \sum_{x \in \mathcal{X}} \xi_x \qquad \text{s.t.}$$

$$\forall x, y:\ S_\mathrm{M}(\{L_t\}, x, y_x) \geq S_\mathrm{M}(\{L_t\}, x, y) + \Delta(y_x, y) - \xi_x. \tag{12}$$

The representer theorem may now be applied independently to each L_t, yielding $L_t^* = M_t \Phi_t^\mathsf{T}$. We define positive semi-definite matrices $W^t = M_t^\mathsf{T} M_t$ specific to each kernel $K^t = \Phi_t^\mathsf{T} \Phi_t$. Similarly, for kernel K^t, let ψ_t^K be as in Eq. 8. Equations 9 and 11 show that, at an optimum, S_M decomposes linearly into kernel-specific inner products:

$$S_\mathrm{M}\left(\{L_t^*\}, x, y\right) = \sum_{t=1}^m \langle W^t, \psi_t^K(x, y) \rangle_\mathsf{F}. \tag{13}$$

We thus arrive at the Multiple Kernel MLR program (MKMLR) listed as Algorithm 3. Algorithm 3 is a linear program over positive semi-definite matrices W^t and slack variables ξ_x, and is therefore convex.

We also note that like the original score function (Eq. 3), S_M is linear in each y_{ij}, so the dependency on y when moving from MLR to MKMLR is essentially unchanged. This implies that the same cutting plane techniques used by MLR—i.e., finding the most-violated constraints—may be directly applied in MKMLR without modification.

Algorithm 3 Multiple Kernel MLR (MKMLR)

Input: Training kernel matrices $K^1, K^2, \ldots, K^m$, true rankings $y_1, y_2, \ldots y_n$, slack trade-off $C > 0$

Output: $n \times n$ matrices $W^1, W^2, \ldots, W^m \succeq 0$

$$\min_{W^t \succeq 0,\ \xi} \quad \sum_{t=1}^m \mathrm{tr}(W^t K^t) + \frac{C}{n} \sum_{x \in \mathcal{X}} \xi_x$$

s.t. $\forall x \in \mathcal{X},\ \forall y \in \mathcal{Y}$:

$$\sum_{t=1}^m \langle W^t, \psi_t^K(x, y_x) \rangle_\mathsf{F} \geq \sum_{t=1}^m \langle W^t, \psi_t^K(x, y) \rangle_\mathsf{F} + \Delta(y_x, y) - \xi_x$$

4 Efficient Region Clustering and Labeling

As the final goal of an object category discovery framework is to fully annotate all unlabeled regions, we include MKMLR into an iterative framework that discovers object categories by efficiently and accurately clustering image regions and queries for cluster annotations.

In order to group together instances that belong to the same object category, we perform agglomerative hierarchical clustering using the optimized distance function d_M. This bottom–up clustering considers the maximum distance between a pair of objects, where each object belongs to a different cluster, to be the distance between these two clusters (complete linkage, Defays 1977). Our decision to use

agglomerative clustering is motivated primarily by its speed and simplicity.

After clustering, we select the prominent cluster c^* with the minimum average intra-cluster distance:

$$d_{\text{intra}}(c) := \frac{1}{|c| \cdot (|c| - 1)} \sum_{\substack{p,q \in c \\ p \neq q}} d_{\text{M}}(x_p, x_q) \tag{14}$$

$$c^* \leftarrow \text{argmin}_c \, d_{\text{intra}}(c). \tag{15}$$

The prominent cluster is then shown to a human annotator, which can either (a) mark the instances that correspond to the largest subset of a particular category; or (b) mark the instances that don't belong to the majority category. Therefore the human always chooses to perform the fewest assignments in order to label the largest group of instances corresponding to a single category.

Figure 4 shows different methods for iterative labeling. In one extreme, the annotator labels one instance in each iteration. This requires the highest level of effort, as the total number of labels queried is equal to the number of unlabeled (initially unfamiliar) instances in the dataset. Alternatively, the system may present a cluster of instances to the annotator and ask for the majority label, which is then applied to all instances in the cluster (Lee and Grauman 2011). This approach greatly reduces annotator effort, but may degrade accuracy due to incorrect labeling of the training data when the clusters are impure.

Our proposed labeling strategy lies between these two extremes, and offers a compromise between accuracy and efficiency. Like the majority vote approach of Lee and Grauman (2011), labeling the majority subset of a cluster allows a single annotation to be applied to many instances, and thus reduces annotator effort compared to the one-at-a-time approach. However, by keeping minority instances unlabeled, we ensure accurate labeling, which will allow for a higher overall classifier accuracy. Table 1 illustrates the main differences between Lee and Grauman (2011) and our method.

Varying the number k of clusters formed in each iteration facilitates a trade-off between the total annotator effort, and per-iteration effort. Large values of k yield smaller, purer clusters, but also provide fewer labels in each iteration, so more iterations are required. Small values of k provide larger, impure clusters that take may take more effort to annotate, but may result in an overall reduction in effort by requiring fewer iterations to label the entire set.

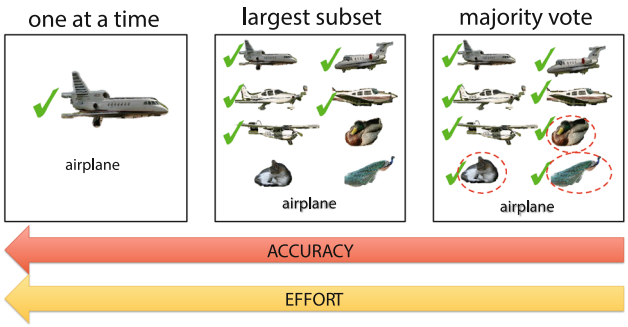

Fig. 4 Comparison between different labeling strategies: (*left*) labeling one instance per iteration, (*middle*) labeling the largest subset of instances belonging to the same class, and (*right*) labeling all instances using the majority class. *Green* check marks show the instances that are annotated in each iteration, and incorrectly labeled instances are circled in *red*. Annotator effort increases when fewer instances are labeled at each iteration, and accuracy decreases when instances are incorrectly labeled (Color figure online)

5 Experiments

In this section, we evaluate the proposed method in two parts. Because similarity learning lies at the core of our framework, we first evaluate the learning algorithm by itself and compare to previous work. Then, we evaluate the entire framework by iteratively updating the similarity metric as new classes are discovered.

In the first set of experiments (Sect. 5.2), we evaluate the metric learning algorithm for a single iteration, according to: (i) the accuracy of segment classification for both familiar and unfamiliar categories, (ii) how well the similarities between intra- and inter-class instances are learned, and (iii) the purity of the clustering performed in the optimized space.

In the second set of experiments (Sect. 5.3), we evaluate the iterative discovery framework by: (i) the efficiency of discovering unfamiliar categories using the native and learned metrics, (ii) the purity of the clustering performed at each iteration, and (iii) the accuracy and label efficiency over time of the entire system on held-out test data.

To evaluate the classification and clustering accuracy of the proposed system, we use the MSRC (Winn et al. 2005) and PASCAL 2007 (Everingham et al. 2007) databases. Our selection of these datasets was motivated by three factors:

1. Both datasets contain at least 20 categories, multiple objects per image, and present challenges such as high intra-class, scale and viewpoint variability.
2. MSRC provides pixel-level ground truth labels for all the objects in the scene, offering more detailed information with which we can evaluate our framework.
3. PASCAL provides ground truth bounding boxes for a few objects in each image, making the problem more difficult in cases where segments with different labels fall inside of the bounding boxes. This makes the evaluation more realistic, as bounding boxes are a popular way of labeling objects for recognition tasks.

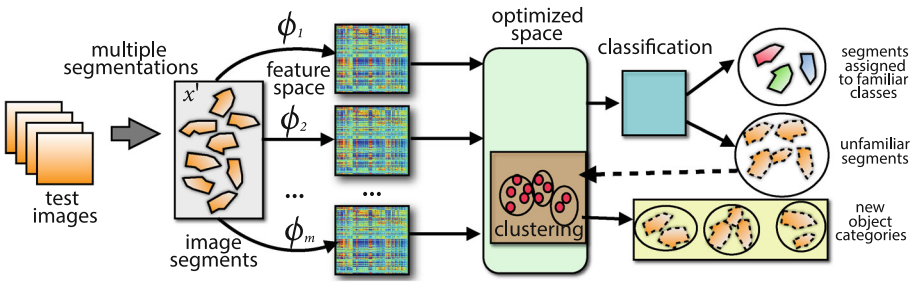

Fig. 5 Discovering object categories: Each test image is partitioned into multiple segments, each of which are mapped into multiple kernel induced feature spaces, and then projected into the optimized similarity space learned by MKMLR (Algorithm 3). Each segment is classified as belonging to a familiar or unfamiliar class by k-nearest-neighbor. Unfamiliar segments are then clustered in the optimized space, enabling the discovery of new categories.

Using ground truth label information (e.g., masks or bounding boxes), each image $\mathcal{I}$ in the image set is partitioned into segments x_i. Each segment x_i belongs to exactly one object of class $\ell_i \in \mathcal{L}$, where $\mathcal{L}$ is the set of familiar object labels.

All segments derived from the set of images are collected to form the region set $\mathcal{X} = \mathcal{X}_\mathrm{m} \cup \mathcal{X}_\mathrm{f} \cup \mathcal{X}_\mathrm{u}$. The set $\mathcal{X}_\mathrm{m}$ contains all training segments x_i derived from ground truth annotations across all images.

Additionally, we partition each image $\mathcal{I}$ into overlapping regions by running a segmentation algorithm multiple times. We use stable segmentations (Rabinovich et al. 2006) to obtain a set of overlapping segments at different scales, where the number of regions per segmentation ranges from 2 to 10. The goal of running several segmentations for the same image is to find scales where objects and parts can be successfully segmented. Only those segments that overlap more than 50 % with a ground truth mask corresponding to a familiar label in $\mathcal{L}$ are collected into the set $\mathcal{X}_\mathrm{f}$. The rest of the segments, which lack (familiar) ground truth labels, are collected in the set $\mathcal{X}_\mathrm{u}$.

Throughout, we will refer to segments corresponding to familiar classes (i.e., $\mathcal{X}_\mathrm{m}$ and $\mathcal{X}_\mathrm{f}$) as *familiar segments*, and segments corresponding to unfamiliar labels ($\mathcal{X}_\mathrm{u}$) as *unfamiliar segments*.[3]

5.1 Features

Six different appearance and contextual features were computed: SIFT, Self-similarity (SSIM), LAB color histogram, PHOG, GIST contextual neighborhoods and LAB color histogram for Boundary Support. For each feature type, we apply an RBF kernel over χ^2-distances, with parameters set to match those reported in (Galleguillos et al. 2010).

[3] *Familiarity* refers to a segment's true label, which may or may not be available: an unlabeled or test segment may be familiar or unfamiliar.

5.2 Evaluation of the Optimized Similarity for a Single Iteration

In order to evaluate the quality of the learned similarity space, we assess the accuracy of predicting first whether or not a test segment is familiar, and if so, its correct label. Therefore, at test time, one-pass object category discovery proceeds as follows (illustrated in Fig. 5):

1. A collection of test images $\mathcal{I}'$ are segmented multiple times to form the test set $\mathcal{X}'$.
2. For each $x' \in \mathcal{X}'$, we use the optimized metric to locate its k-nearest neighbors from the training set.
3. The k-nearest training segments are used to estimate a distribution over the labels $P(\ell|x')$, where $\ell \in \mathcal{L} \cup \{\ell_0\}$, and ℓ_0 is a synthetic label given to all unlabeled training segments $\mathcal{X}_\mathrm{u}$. A segment is assigned to its most probable category:

$$\ell(x') = \operatorname*{argmax}_{\ell \in \mathcal{L} \cup \{\ell_0\}} P(\ell|x').$$

4. After classifying each $x' \in \mathcal{X}'$, all segments with predicted label ℓ_0 are used as input to a clustering algorithm.
5. We use spectral clustering (Meila and Shi 2001)[4] with affinities defined by a radial basis function (RBF) kernel on the learned distances:

$$A_{ij} = \exp\left(-d_\mathrm{M}(x'_i, x'_j)/2\sigma^2\right),$$

where $d_W(x'_i, x'_j)$ is the squared (optimized) distance between two test segments x'_i and x'_j, and σ is a bandwidth parameter.

[4] We chose spectral clustering over agglomerative clustering in this set of experiments to facilitate direct comparison to Lee and Grauman (2010).

Table 3 Partitions for unfamiliar and familiar classes for (a) MSRC and (b) PASCAL VOC 2007.

	Set	**Familiar**	class #s	**Unfamiliar**	class #s
(a)		Bird, boat, body, book, car			
	1	cat, chair, dog, face, flower, grass road, sheep, sign, sky, water	3–6, 8–10, 12–19, 21	Aeroplane, bicycle, building, tree, cow	1, 2, 7, 11, 20
	2	Body, book, building, car, cat, cow dog, face, flower, grass, sign	5–9, 11–15, 18	Aeroplane, bicycle, bird, boat, chair road, sheep, sky, tree, water	1–4, 10, 16, 17, 19–21
	3	Car, dog, flower, grass, tree, water	8, 12, 14, 15, 20, 21	Aeroplane, bicycle, bird, boat, body, book, building cat, chair, cow, face, road, sheep, sign, sky	1–7, 9–11, 13, 16–19
(b)		Bicycle, boat, bottle, bus, car, cat			
	1	chair, dining table,dog, horse, person potted plant, sheep, sofa, train	2, 4–9, 11–13, 15–19	Airplane, bird, cow, motorbike, tv	1, 3, 10, 14, 20
	2	Bicycle, bird, car, cat cow, dog, horse, potted plant, tv	2, 3, 7, 8, 10, 12, 13, 16, 20	Airplane, boat, bottle, bus, chair, cow, motorbike person, sheep, sofa, train	1, 4–6, 9, 11, 14, 15, 17–19
	3	Airplane, bicycle, bird, person, sheep	1–3, 15, 17	Boat, bottle, bus, car, cat, chair, cow, dining table dog, horse, motorbike, potted plant, sofa, train, tv	4–14, 16, 18–20

Table 4 The number of known categories ($\mathcal{L}$) and training and test segments in each partition of the datasets. $\mathcal{X}_m$ are the training regions obtained from ground truth segmentations (MSRC) and ground truth bounding boxes (PASCAL). $\mathcal{X}_f$ are the familiar training segments, $\mathcal{X}_u$ are the unlabeled segments, X'_f are the familiar testing segments and $\mathcal{X}'_u$ are the unlabeled testing segments (all generated by automatic segmentation)

	MSRC			PASCAL		
	Set 1	Set 2	Set 3	Set 1	Set 2	Set 3
$\lvert\mathcal{L}\rvert$	16	11	6	15	10	5
$\lvert\mathcal{X}_m\rvert$	640	548	322	458	278	174
$\lvert\mathcal{X}_f\rvert$	870	583	318	535	321	183
$\lvert\mathcal{X}_u\rvert$	261	435	813	180	394	532
$\lvert\mathcal{X}'_f\rvert$	4124	3160	2375	583	330	206
$\lvert\mathcal{X}'_u\rvert$	1975	2939	3724	200	453	577

For experiments with MSRC, we use the same train and test split as Lee and Grauman (2010) (hereafter referred to as LG10), and the object detection split of PASCAL VOC 2007 (Everingham et al. 2007). We adopt three different partitionings of each dataset into unfamiliar/familiar classes from LG10 for comparison purposes. The different class partitions are shown in Table 3 and statistics of each partition are reported in Table 4.

Note that the number of examples in PASCAL VOC 07 is smaller than in MSRC. This is because PASCAL images may contain unlabeled regions[5], and few objects are labeled in each image. Training segmentations were sub-sampled in order to preserve balance within the training set with respect to the bounding box regions. We retain only the largest two segments per object in each image.

[5] Weak labeling in PASCAL dataset makes it difficult to evaluate due to background segments without ground truth.

5.2.1 Classification of Unfamiliar Segments

We evaluate the learned similarity space by computing classification accuracy over the full test set ($\mathcal{X}'_f \cup \mathcal{X}'_u$). For each partition (Set 1, 2, 3) of MSRC and PASCAL, we train a metric with MKMLR on the entire training set. For comparison purposes, we repeat the experiment on metrics learned by MKLMNN, as well as the "native" feature spaces formed by taking the unweighted combination of base kernels. At test time, a segment is predicted to belong either to one of the familiar classes in $\mathcal{L}$, or the *unfamiliar* class ℓ_0. The overall accuracy is reported in Table 5.

When there are fewer familiar classes from which to choose, the problem becomes easier because more test segments must belong to the *unfamiliar* class. This trend is demonstrated by the increasing accuracy of each algorithm from Set 1 (5 unfamiliar classes) to Set 2 (10 unfamiliar) and Set 3 (15 unfamiliar).

In MSRC, where image regions are densely labeled, we observe that MKMLR consistently outperforms MKLMNN and the native space, although the gap in performance is largest when more supervision is provided. In PASCAL, however, we observe that the unweighted kernel combination achieves the highest accuracy for Sets 2 and 3, i.e., the sets with the least supervision. This may be attributed to MKLMNN and MKMLR over-fitting the training set, which for PASCAL is considerably smaller than that of MSRC (see Table 4). The over-fitting in question is primarily an artifact of using a weakly labeled dataset (PASCAL07) to simulate the annotation process in our experiments. In a live setting with human annotators, all segments (including background) may receive categorical annotations, as in the MSRC setting.

5.2.2 Intra-Class Versus Inter-Class Affinities

Our second evaluation replicates an experiment on MSRC Set 1 in LG10 (Table 1, Lee and Grauman (2010)), and measures the accuracy of the learned distance function independent of clustering. A distance matrix is computed for all pairs of test segments predicted to be unfamiliar by the segment classification step, and for each test segment, the remaining segments are ranked by increasing distance. Then, using the ground-truth labels, the average precision of each ranking is computed for each test segment, and the resulting scores are combined to form a mean average precision (MAP) score.

Relying on the segment classification step to determine which points are familiar and unfamiliar may introduce bias to the evaluation. We therefore repeat the above experiment using ground-truth familiar and unfamiliar labels. Table 6 shows the MAP results by first classifying the regions between familiar and unfamiliar. Table 7 shows the MAP results assuming perfect familiar/unfamiliar classification. For completeness, we again compare the performance of MKMLR to MKLMNN[6] as well as Kernel MLR (KMLR) on the unweighted average kernel $\widehat{K} := \sum_t K^t$.

We observe in the unbiased evaluation (Table 7) that MKMLR outperforms the other methods under consideration for all categories. We also note that in the self-biased evaluation (Table 6), MKMLR achieves significantly higher familiar/unfamiliar accuracy than the alternative methods.

5.2.3 Cluster Purity

Our third evaluation concerns the purity of clusters discovered in the test data. Again, we compare the native (unweighted) kernel combination, MKLMNN, and MKMLR on each partition of MSRC and PASCAL. For each set, we replicate the experiment of LG10 (Lee and Grauman 2010, Fig. 5), and using the ground-truth labels, perform spectral clustering in the optimized space on the test segments belonging to unfamiliar classes. We vary the number of clusters from 2 to 35, and for each of them, compute the average *purity* (Zhao and Karypis 2001) of the clustering, where a cluster B's purity is defined as

$$\text{purity}(B) := \max_{\ell \in \mathcal{L}} \frac{\left|\{x' \in B \wedge \ell(x') = \ell\}\right|}{|B|}.$$

For each number of clusters, we generate 10 different clusterings, and report the average purity. The resulting mean purity curves are reported in Fig. 6.

We observe that in almost all cases, the mean purity achieved by MKMLR lies (significantly) above that of the native space, and is often significantly above that achieved by MKLMNN.

The reduced purity scores for PASCAL (relative to MSRC) can be attributed to two facts. First, the sparsity of ground truth labels in PASCAL indicates that the evaluation here is somewhat less thorough than for MSRC. Second, as described in Sect. 5.2.1, the reduced size of the training set leads to some over-fitting by both MKLMNN and MKMLR. However, we stress that the reduction in performance is largely an artifact of sparse ground-truth annotation, which would be overcome in practice by dense annotations produced by the proposed interactive evaluation scheme with a human annotator.

Table 5 Nearest-neighbor classification accuracy of MKMLR, MKLMNN, and the native feature space, including the *unfamiliar* class ℓ_0

	Algorithm	Set 1	Set 2	Set 3
MSRC	Native	0.51	0.59	0.71
	MKLMNN	0.61	0.57	0.69
	MKMLR	**0.62**	**0.61**	**0.72**
PASCAL07	Native	0.31	**0.58**	**0.74**
	MKLMNN	0.32	0.51	0.67
	MKMLR	**0.33**	0.54	0.70

The best scores are in bold

5.3 Iterative Discovery

In this section, we evaluate the annotator efficiency and predictive accuracy of the full iterative category discovery system.

5.3.1 Efficient Iterative Clustering

While Lee and Grauman (2011) also address iterative category discovery, their experimental evaluation focuses primarily on clustering and segmentation accuracy. Our goal differs

[6] In Table 6, MKLMNN (Galleguillos et al. 2010) has no MAP score for class *tree* because there was only one test segment of that class predicted as unfamiliar.

Table 6 Comparison of MAP scores for MSRC Set 1 for segments predicted to be unfamiliar. Acc_f and Acc_u are the average accuracy of predicting segments as familiar and unfamiliar respectively

Methods	Airplane	Bicycle	Building	Cow	Tree	Acc_f	Acc_u
LG10	0.36	0.21	0.32	0.41	0.36	–	–
Native	0.61	0.37	0.30	0.40	0.55	0.68	0.40
MKLMNN	0.75	0.51	**0.38**	**0.71**	–	0.85	0.42
KMLR (avg.)	0.66	0.45	0.35	0.40	0.49	0.81	0.45
MKMLR	**0.84**	**0.58**	**0.38**	0.41	**0.70**	**0.95**	**0.54**

The best scores are in bold

slightly, in that we are primarily concerned with the efficiency of acquiring accurate labels (Table 1). We therefore evaluate our framework in terms of the number of iterations required to label all regions. We report performance in terms of the relative improvement over the worst case scenario, in which each image region is labeled independently. Fewer iterations means that more labels are acquired in each iteration, thereby reducing annotator effort.

To evaluate the accuracy and efficiency of our iterative framework, we simulate a human annotator that uses ground-truth labels to correctly annotate the largest subset of a cluster.

Different data sets containing different numbers of familiar (seed) object categories are generated for these experiments. These sets consist of between one and eight different randomly chosen object categories, and for each set we generate 18 different random sets of familiar/unfamiliar classes. The number of clusters computed in each iteration is fixed to 10, and the number of neighbors used for learning the metric is 15 in all experiments. Object categories considered in our experiments belong to the MSRC image database, which is composed of 21 different classes. In the terminology of Heitz and Koller, these classes can be grouped into two categories: *things* and *stuff* (Heitz and Koller 2008). *Things* refer to discrete, man-made, or rigid objects—e.g., a person, car, chair, etc.—that can be identified from the appearance in a small region around the object. *Stuff* refers to regions of amorphous spatial extent that are more naturally classified based on texture or color, such as sky, grass, or water. The images used in the experiments correspond to the training data split used in previous experiments.

Figure 7 shows the efficiency of our framework when discovering categories iteratively on the randomly generated sets. We compare our framework (learned metric) to the native metric by considering sets composed of different numbers of seed classes. Each curve corresponds to the average percentage of iterations required to label all unfamiliar instances. We observe that on average we need a smaller percentage (<6 % of the total iterations to label all regions. The worst case involves using 100 % of the iterations—i.e., labeling only one instance per query—and the best case possible would to require <1%—i.e., labeling all instances of a category per query.

We also observe that the learned metric significantly outperforms the native metric across all numbers of seed classes, indicating that improving the underlying similarity metric can indeed reduce the human annotator effort. However, even without optimizing the similarity metric, the interactive annotation model still provides substantial benefits over one-at-a-time labeling. In general, for both learned and native metrics, the percentage of iterations required to label all instances decreases with the number of initial seed classes. We compare the number of iterations of the two metrics using the Wilcoxon sign-rank test (Wilcoxon 1945). The largest p value for all sets is $1.9609e-04$ (for 8 initial seeds).

From these experiments, we identify the set of classes that deliver the best and worst percentage of iterations for a given number of initial seeds.

Table 8 shows examples of worst and best cases found in our experiments using the native metric, as well as using our framework (learning the similarity metric). Class numbers in bold correspond to *stuff* categories. We observe that worst cases in the native space are mostly composed by *things* rather than *stuff* categories, meanwhile there is no obvious correlation for those using the learned metric. This indicates that our framework is unbiased to the type of category (*stuff* vs. *things*) that is used as initial set in order to efficiently label all instances.

Table 7 Comparison of MAP scores for MSRC Set 1 on true unfamiliar segments

Methods	Airplane	Bicycle	Building	Cow	Tree
Native	0.65	0.43	0.33	0.36	0.57
MKLMNN	0.68	0.50	0.44	0.59	0.59
KMLR (avg.)	0.72	0.46	0.38	0.46	0.59
MKLMLR	**0.81**	**0.55**	**0.45**	**0.71**	**0.66**

The best scores are in bold

5.3.2 Accurate Iterative Clustering

In order to assess the predictive accuracy of the iterative annotation framework, we evaluate the average purity of a discovered category using the randomly generated sets described in the previous section. Table 9 shows the average purity of a new discovered category given the number of seed classes. The average purity corresponding to the learned metric is usually higher for most of the sets than using the native space

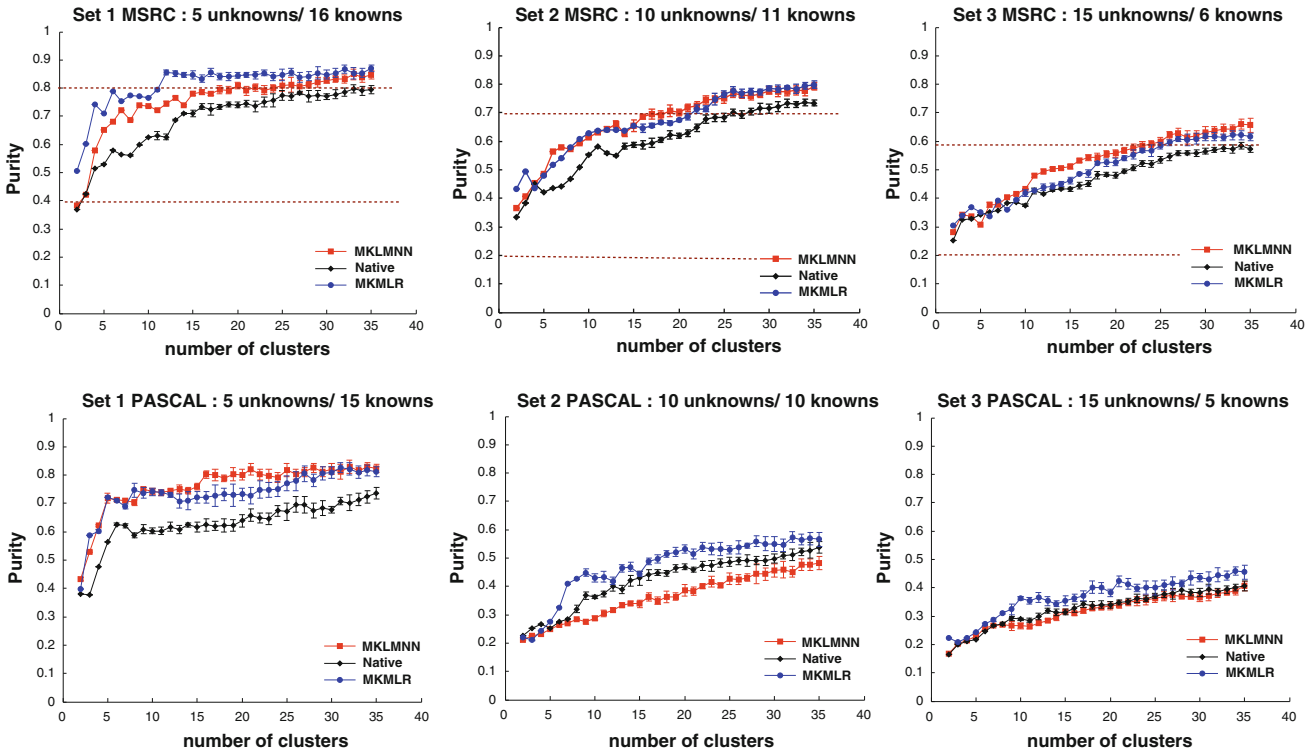

Fig. 6 Mean cluster purity curves. *Top* plots correspond to different sets in MSRC, and *bottom* plots correspond to PASCAL VOC2007. *Error bars* correspond to one standard deviation. *Dashed lines* correspond to bounds on purity scores reported by LG10 (Fig. 5e, Lee and Grauman (2010))

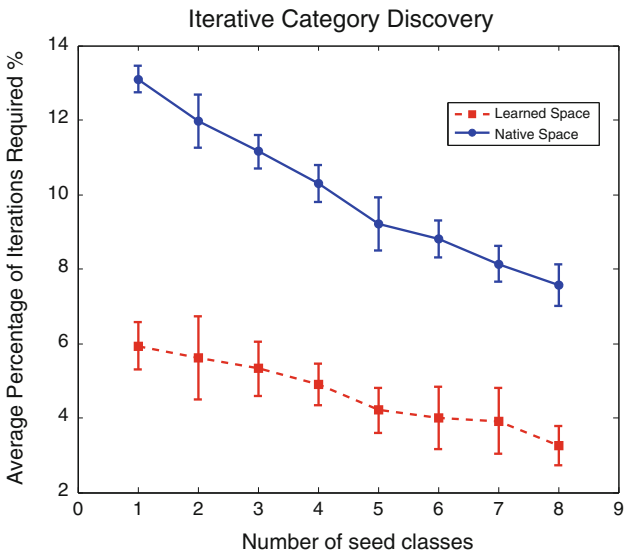

Fig. 7 Percentage of total iterations required to label all instances versus the number of initial familiar classes.(Lower is better). *Error bars* are shown for the mean curves

(Table 9), and tends to increase with the number of initial seed classes. The optimized metric outperforms the native metric as we obtain comparable (or better) average purity across all sets of initial seeds classes but with much fewer queries required on average (as shown in Fig. 7).

As the average purity increases with respect to the different number and type of seed sets when optimizing the space, we investigate how efficiency and purity are affected when using different numbers of clusters in the grouping step of our algorithm (Algorithm 1). Figure 8 illustrates different results obtained using 10, 20 and 30 clusters per iteration for all sets. Figure 8a depicts the mean purity of a discovered class obtained using our framework. Error bars show that cluster purity can vary around 10 % for different number of initial classes when using different numbers of clusters. The results demonstrate that increasing the number of clusters improves the purity of the discovered clusters. The number of clusters that offers the best purity without compromising on efficiency is found to be around 20 clusters.

Figure 8b presents the efficiency of our method measured by the percentage of total iterations and number of iterations respectively. In the case of 30 clusters per iteration, our framework is still extremely efficient as it executes $< 20\,\%$ of the total iterations needed to label all instances. While the number of iterations decreases when using more seed classes, it becomes clear that the difference between using 30 and 20 clusters is small (with respect to the total) and therefore the trade off between purity and efficiency becomes evident when choosing the right number of clusters.

Table 8 Average percentage of total iterations required to label all instances using the native and learned metric respectively

Native metric N	Best case Object categ.	%	Worst case Object categ.	%
1	5	12.69	1	13.76
2	5,**19**	10.64	1,17	13.31
3	9,10,**21**	10.44	17,**19**,**21**	11.94
4	7,10,13,**19**	9.65	1,4,6,18	11.61
5	3,7,**15**,**16**,18	7.86	2,4,9,12,17	10.54
6	5,7,13,**15**,**16**,**19**	8.04	1,3,4,5,8,11	9.99
7	9,12,**15**,**16**,17,**20**,**21**	7.23	4,5,9,10,13,18,**21**	9.02
8	3,5,7,12,13,**16**,**20**,**21**	6.36	1,2,5,6,8,11,12,17	8.43
Learned metric N	**Best case Object categ.**	**%**	**Worst case Object categ.**	**%**
1	**20**	5.16	12	7.33
2	15,**20**	3.86	1,10	7.05
3	4,**19**,**21**	4.20	10,12,18	7.21
4	1,4,**16**,**20**	4.19	6,8,11,**21**	6.30
5	3,7,**15**,**16**,18	3.32	3,10,11,12,**21**	5.46
6	8,9,13,**15**,17,**20**	2.64	1,6,7,8,12,**14**	5.95
7	9,12,**15**,**16**,17,**20**,**21**	2.54	1,6,10,11,**14**,**19**,**21**	5.62
8	1,4,5,9,12,13,17,**21**	2.60	2,3,10,11,12,**14**,17,**20**	4.48

Examples of (a) best and (b) worst cases considering different number of seed classes Class numbers in bold correspond to *stuff* categories

Table 9 Average purity for native and learned metric

Metric	1	2	3	4	5	6	7	8
Native	**0.49**	0.49	0.52	0.53	0.57	0.53	**0.56**	0.62
Learned	0.48	**0.54**	**0.58**	**0.61**	**0.58**	**0.60**	0.55	**0.63**

Mean purity of a discovered cluster is shown for each set of initial seed classes
The best scores are in bold

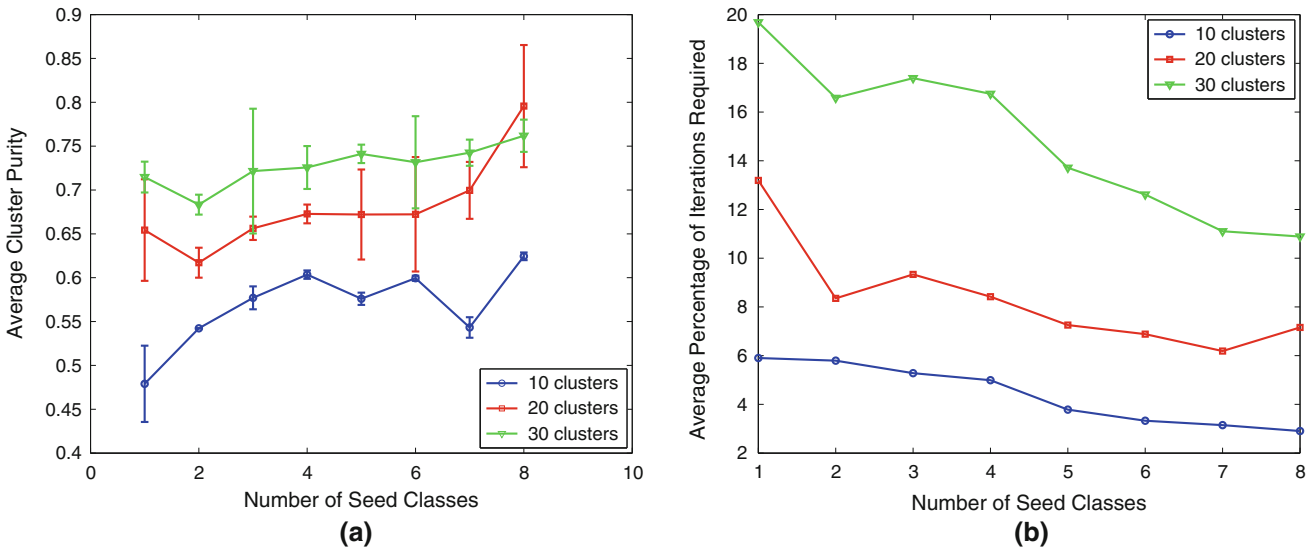

Fig. 8 Experimental results for our framework using 10, 20 and 30 clusters per iteration. **a** Mean purity results of a discovered class with *error bars* for the mean curves, **b** percentage of total iterations. Experiments are computed using the randomly generated familiar/unfamiliar sets described in 5.3.1

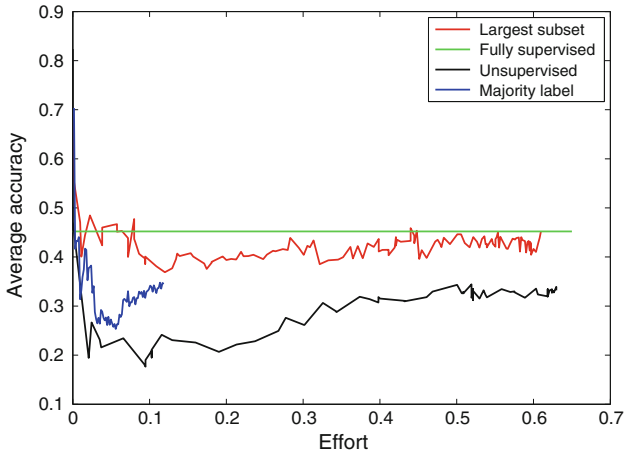

Fig. 9 Average classification accuracy on held-out instances versus the effort required by the human annotator. Accuracy at each step computed by considering only the currently discovered categories. Discovered categories are labeled using the majority vote and the largest subset approach

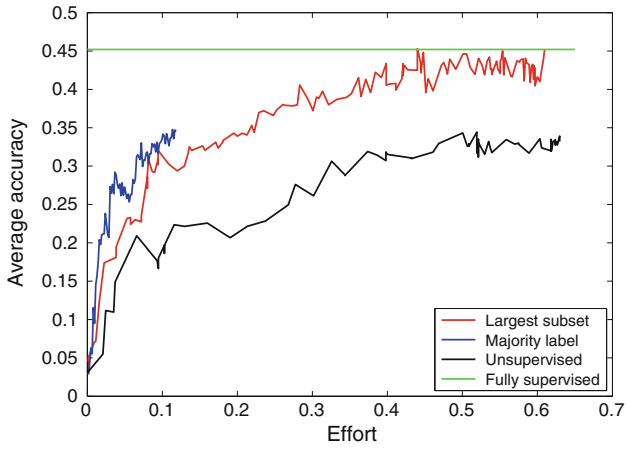

Fig. 10 Average classification accuracy on held-out instances versus the effort needed to label instances in the training set. Accuracy is computed by considering all 16 object classes at each iteration, including those yet to be discovered

5.3.3 Accuracy Versus Effort

Our final experiment measures the effort associated with labeling the discovered classes at each iteration, and how the accuracy of the system improves over time. We replicated the experimental setup of Lee and Grauman (2011) to evaluate label efficiency and classifier accuracy as follows. The MSRC data was first randomly partitioned 40/60. From the 40 % partition, we extracted five initial *stuff* seed categories: *grass*, *sky*, *road*, *tree* and *water*. Given those seed instances, the system is then allowed to discover the remaining unfamiliar categories on the 60 % subset. At each iteration we measure the classification accuracy of the remaining *unfamiliar* instances from the 40 % subset; note that these instances are merely used for evaluation, and do not influence the category discovery algorithm. Unlike Lee and Grauman (2011), we do not use any external data to train classifiers to generate candidate segments.

At each iteration of category discovery, the most prominent cluster is labeled, and all labeled instances are used to learn a similarity metric.

Once the metric has been learned, we evaluate nearest-neighbor classification accuracy on the held-out instances, and record the mean accuracy. We measure the effort of labeling instances as the number of "clicks" performed by the user to label the largest subset, divided by the total number of instances. For comparison purposes, we also test the labeling strategy of Lee and Grauman (2011) where each cluster is entirely labeled by its majority label.

We also simulate an unsupervised baseline, in which the selected cluster at each iteration is completely assigned a new class label with no human annotator intervention, so the number of classes discovered is equal to the number of iterations.

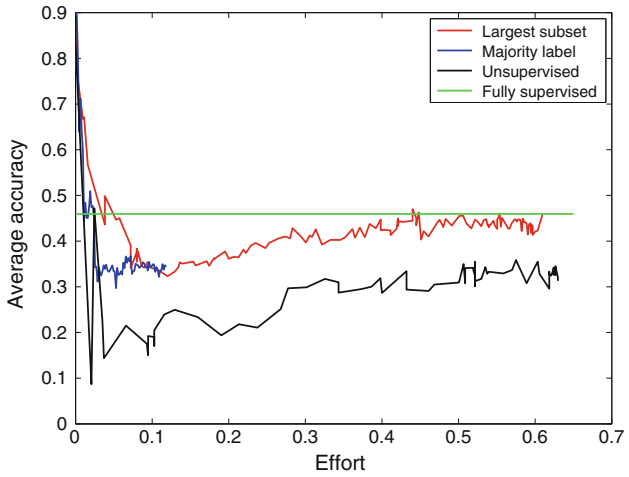

Fig. 11 Average classification accuracy on held-out instances versus the effort needed to label instances in the training set. Accuracy is computed by considering segments to belong to a familiar or unfamiliar class at each iteration

At each iteration, the metric is learned using the newly discovered class, and classification accuracy is computed using the ground truth labels.

Figure 9 shows the average familiar-category accuracy achieved on held-out data versus the effort demanded by the human annotator for both labeling strategies, as well as the fully supervised and the unsupervised baseline. While the majority label strategy terminates earlier, the incorrect annotations negatively impact accuracy, and it is quickly dominated by the more conservative, largest-subset labeling strategy.

Figure 10 shows the accuracy across all categories, including those which have not yet been discovered (in

which case, the system should predict ℓ_0/unfamiliar). Again, the majority-label strategy is more efficient in terms of effort, but since it cannot revise potentially mislabeled data, its accuracy is limited. Moreover, the majority-labeling strategy is not guaranteed to discover categories which never become the majority of a cluster. In contrast, the largest-subset strategy eventually matches and overtakes the accuracy of majority-label while still only using a fraction of the labels, and converges to the fully supervised case.

Figure 11 shows the accuracy of classifying segments to be familiar or unfamiliar (ℓ_0). Similarly to Fig. 9, the largest-subset strategy outperforms the accuracy of majority-label and converges to the fully supervised case. As expected, the accuracy of the unsupervised framework remains more than 10 % below of our framework's accuracy.

These figures illustrate that the proposed method achieves higher prediction accuracy than the majority-label approach, and converges to the accuracy achieved by the fully-labeled setting. The unsupervised and majority label approaches do not converge to the fully supervised case.

Figure 12 show examples of segments discovered by our framework at certain iterations. Our automatic segmentation (Rabinovich et al. 2006) tends to over segment the image, which is reflected in some of these examples (specially iteration 106). We observe less variability in the segments discovered within the initial iterations, while they become more visually diverse when the number of iterations increases. In addition, we see that the number of segments in the clusters decreases in later iterations. Segments in iteration 9 (airplanes) contain small parts of other classes (road, building), which is one of the shortcomings of the automatic segmentation. However, a sufficient portion of the object's area is contained within the segment to yield accurate clustering. Alternatively, iterations 1, 17 and 36 have segments that contain only one object class, and only minimal parts of the object are missing. Segments in iteration 106 (bicycles) are primarily over-segmentations, but may also span multiple objects.

Fig. 12 Examples of categories discovered at each iteration. The number before the image set indicates the iteration in which the segments received their annotation. Classes discovered: (*1*) flower, (*9*) aeroplane, (*17*) face, (*36*) building, (*80*) sign, (*106*) bicycle

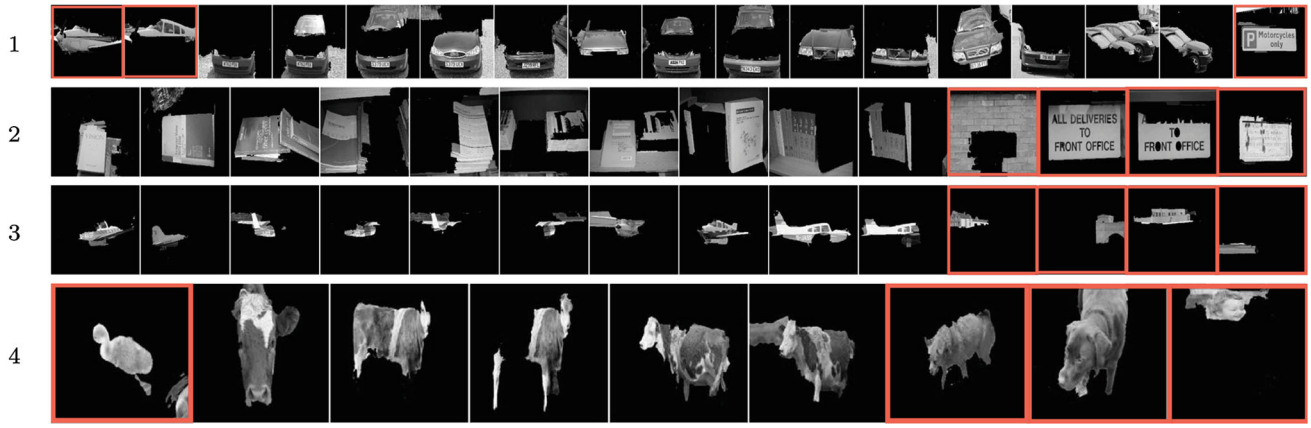

Fig. 13 Examples of segment clusters, annotated with majority-subset membership. The discovered classes are: (*1*) car, (*2*) book, (*3*) aeroplane and (*4*) cow. Segments highlighted in *(red)* belong to minority categories, and are not labeled in the iteration (Color figure online)

Nonetheless, the resulting segments are still visually similar, and they are correctly grouped together under the learned metric.

Examples of clusters with human annotations are shown in Fig. 13. Each cluster contains a predominant class, which represent more than 70 % of the segments. Cluster 1 contains mostly cars and car parts, and only 3 segments out of 17 correspond to different classes (airplanes, sign). We observe that the outliers in this cluster are visually similar with the segments belonging to main class. Similarly, the segments in clusters 2 and 4 present visually similar information, however only segments belonging to the largest subset (categories book and cow) are annotated in this iteration. Cluster 3 shows examples of segments generated by the automatic segmentation corresponding to small objects (parts of planes and buildings). Given the limited information that these segments carry, they are difficult to discover within the same iteration. Our framework successfully groups these segments together and labels them as the same category.

6 Conclusion

We have introduced a novel framework for iterative object class discovery. The proposed method iteratively discovers new object categories by efficiently and accurately grouping image regions under the optimized distance metric. Our experimental results demonstrate that improving region similarity can greatly reduce human annotation efforts: the human annotator needs only to supply a fraction of labels to achieve comparable accuracy to the fully annotated setting, and the proposed annotator interaction model maintains the accuracy of training data.

Appendix: Implementation

The implementation uses the 1-slack margin-rescaling cutting plane algorithm (Joachims et al. 2009) to solve for all W^t within a prescribed tolerance $\epsilon = 0.01$. We further constrain each W^t to be a diagonal matrix. This simplifies the semi-definite program to a linear program. For m kernels and n training points, this also reduces the number of parameters needed to learn from $O(mn^2)$ (m symmetric n-by-n matrices) to mn.

In all experiments with MKMLR, we choose the ranking loss Δ as the normalized discounted cumulative gain (NDCG) (Järvelin and Kekäläinen 2002) truncated at 10. Slack parameters C and kernel bandwidth σ for spectral clustering were found by cross-validation on the training set. For testing, we fix $k = 17$ as the number of nearest neighbors for classification across all experiments. Multiple stable segmentations were computed—9 different segmentations for each image—each of which contains between 2 and 10 segments, resulting in 54 segments per image (Rabinovich et al. 2006; Shi and Malik 2000).

References

Bart, E., Porteous, I., Perona, P., & Welling, M. (2008). Unsupervised learning of visual taxonomies. In *Computer vision and pattern recognition (CVPR)* (pp. 1–8).

Branson, S., Wah, C., Schroff, F., Babenko, B., Welinder, P., Perona, P., et al. (2010). Visual recognition with humans in the loop. In *European conference in computer vision (ECCV)* (pp. 438–451)

Collins, B., Deng, J., Li, K., & Fei-Fei, L. (2008). Towards scalable dataset construction: An active learning approach. In *Computer Vision—ECCV*.

Cortes, C., & Vapnik, V. (1995). Support-vector networks. *The Journal of Machine Learning Research*, *20*(3), 273–297.

Defays, D. (1977). An efficient algorithm for a complete link method. *The Computer Journal*, *20*(4), 364–366.

Everingham, M, Van Gool, L, Williams, CKI, Winn, J, Zisserman, A (2007). *The PASCAL visual object classes, challenge 2007 (VOC2007) Results*.

Faktor, A., & Irani, M. (2012). "Clustering by composition"—unsupervised discovery of image categories. In *European conference in computer vision (ECCV)* (pp. 474–487). Springer.

Forsyth, D. A., Malik, J., Fleck, M. M., Greenspan, H., Leung, T., Belongie, S., et al. (1995). Finding pictures of objects in large collections of images. *The Computer Journal*, *1144*, 335–360.

Frome, A., Singer, Y., Sha, F., & Malik, J. (2007). Learning globally-consistent local distance functions for shape-based image retrieval and classification. In *International conference in computer vision (ICCV)* (pp. 1–8).

Galleguillos, C., McFee, B., Belongie, S., & Lanckriet, G. (2010). Multi-class object localization by combining local contextual interactions. *Computer vision and pattern recognition (CVPR)* (pp. 113–120).

Galleguillos, C., McFee, B., Belongie, S., & Lanckriet, G. (2011). From region similarity to category discovery. In *Computer vision and pattern recognition (CVPR)* (pp. 2665–2672).

Gehler, P., & Nowozin, S. (2009). On feature combination for multiclass object classification. In *International conference in computer vision (ICCV)*.

Globerson, A., & Roweis, S. (2007). Visualizing pairwise similarity via semidefinite embedding. In *International conference on artificial intelligence and statistics (AISTATS)*.

Grauman, K., & Darrell, T. (2006). Unsupervised learning of categories from sets of partially matching image features. In *Computer vision and pattern recognition (CVPR)*.

Heitz, G., & Koller, D. (2008). Learning spatial context: Using stuff to find things. In *European conference in computer vision (ECCV)* (pp. 30–43). Springer:In.

Järvelin, K., & Kekäläinen, J. (2002). Cumulated gain-based evaluation of IR techniques. *ACM Transactions on Information Systems*, *20*(4), 422–446.

Joachims, T. (2005). A support vector method for multivariate performance measures. In *International conference onmachine learning* (pp. 377–384).

Joachims, T., Finley, T., & Yu, C. N. J. (2009). Cutting-plane training of structural svms. *The Journal of Machine Learning Research*, *77*(1), 27–59.

Kang, H., Hebert, M., Efros, A. A., & Kanade, T. (2012). Connecting missing links: object discovery from sparse observations using 5 million product images. *European conference in computer vision (ECCV)* (pp. 794–807). Springer.

Lanckriet, G. R. G., Cristianini, N., Bartlett, P., El Ghaoui, L., & Jordan, M. I. (2004). Learning the kernel matrix with semidefinite programming. *The Journal of Machine Learning Research*, *5*, 27–72.

Lee, Y., & Grauman, K. (2010). Object-graphs for context-aware category discovery. In *Computer vision and pattern recognition (CVPR)*.

Lee, Y., & Grauman, K. (2011). Learning the easy things first: Self-paced visual category discovery. In *Computer vision and pattern recognition (CVPR)* (pp. 1721–1728).

McFee, B., & Lanckriet, G. (2010). Metric learning to rank. In *International conference on machine learning*.

Meila, M., & Shi, J. (2001). Learning Segmentation by Random Walks. *Advances in neural information processing systems*.

Rabinovich, A., Lange, T., Buhmann, J., & Belongie, S. (2006). Model order selection and cue combination for image segmentation. In *Computer vision and pattern recognition (CVPR)*.

Russell, B., Freeman, W., Efros, A., Sivic, J., & Zisserman, A. (2006). Using multiple segmentations to discover objects and their extent in image collections. In *Computer vision and pattern recognition (CVPR)*.

Schölkopf, B., Herbrich, R., Smola, A. J., & Williamson, R. (2001). A generalized representer theorem. In *Computational learning theory* (pp. 416–426).

Shi, J., & Malik, J. (2000). Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *22*(8), 888–905.

Sivic, J., Russell, B., Efros, A., Zisserman, A., & Freeman, W. (2005). Discovering objects and their location in images. In *International conference in computer vision (ICCV)*.

Sivic, J., Russell, B., Zisserman, A., Freeman, W., & Efros, A. (2008). Unsupervised discovery of visual object class hierarchies. In *Computer vision and pattern recognition (CVPR)* (pp. 1–8).

Tian, Y., Liu, W., Xiao, R., Wen, F., & Tang, X. (2007). A face annotation framework with partial clustering and interactive labeling. In *Computer vision and pattern recognition (CVPR)* (pp. 1–8).

Todorovic, S., & Ahuja, N. (2006). Extracting subimages of an unknown category from a set of images. In *Computer vision and pattern recognition (CVPR)*.

Tsochantaridis, I., Joachims, T., Hofmann, T., & Altun, Y. (2005). Large margin methods for structured and interdependent output variables. *The Journal of Machine Learning Research*, *6*, 1453–1484.

Tuytelaars, T., Lampert, C., Blaschko, M., & Buntine, W. (2010). Unsupervised object discovery: A comparison. *International Journal of Computer Vision*, *88*(2), 284–302.

Varma, M., & Ray, D. (2007). Learning the discriminative power-invariance trade-off. In *International conference in computer vision (ICCV)*.

Vedaldi, A., Gulshan, V., Varma, M., & Zisserman, A. (2009). Multiple kernels for object detection. In *International conference in computer vision (ICCV)*.

Vijayanarasimhan, S., & Grauman, K. (2009). What's it going to cost you? Predicting effort vs. informativeness for multi-label image annotations. In *Computer vision and pattern recognition (CVPR)*.

Wang, G., Hoiem, D., & Forsyth, D. (2010). Learning image similarity from flickr groups using stochastic intersection kernel machines. In *Computer vision and pattern recognition (CVPR)*.

Weinberger, K. Q., Blitzer, J., & Saul, L. K. (2006). Distance metric learning for large margin nearest neighbor classification. *Advances in neural information processing systems*.

Wilcoxon, F. (1945). Individual comparisons by ranking methods. *Biometrics Bulletin*, *1*(6), 80–83.

Winn, J., Criminisi, A., & Minka, T. (2005). Object categorization by learned universal visual dictionary. In *International conference in computer vision (ICCV)* (Vol. 2, pp. 1800–1807).

Zhao, Y., & Karypis, G. (2001). Criterion functions for document clustering: Experiments and analysis. *Machine Learning*.

Zhu, J. Y., Wu, J., Wei, Y., Chang, E., & Tu, Z. (2012). Unsupervised object class discovery via saliency-guided multiple class learning. In *Computer vision and pattern recognition (CVPR)* (pp. 3218–3225).