From Region Similarity to Category Discovery

Carolina Galleguillos[†] Brian McFee[†] Serge Belongie[†] Gert Lanckriet[‡]
[†]Computer Science and Engineering Department
[‡]Electrical and Computer Engineering Department
University of California, San Diego
{cgallegu,bmcfee,sjb}@cs.ucsd.edu, gert@ece.ucsd.edu

Abstract

The goal of object category discovery is to automatically identify groups of image regions which belong to some new, previously unseen category. This task is typically performed in a purely unsupervised setting, and as a result, performance depends critically upon accurate assessments of similarity between unlabeled image regions.

To improve the accuracy of category discovery, we develop a novel multiple kernel learning algorithm based on structural SVM, which optimizes a similarity space for nearest-neighbor prediction. The optimized space is then used to cluster unlabeled data and identify new categories.

Experimental results on the MSRC and PASCAL VOC2007 data sets indicate that using an optimized similarity metric can improve clustering for category discovery. Furthermore, we demonstrate that including both labeled and unlabeled training data when optimizing the similarity metric can improve the overall quality of the system.

1. Introduction

The design of accurate models for large collections of object categories has become a central goal of object recognition research. In recent years, the predominant approach to tackling this problem has been to collect labeled examples of each category, which are then provided as input to a machine learning algorithm. When only a relatively small number of categories are to be learned, this general approach performs quite well. However, as the number of categories increases, the acquisition of a sufficiently large and accurate set of training examples becomes an expensive and time-consuming chore. As a result, much research has been devoted to designing efficient schemes for collecting training data for supervised object recognition [2, 3, 24].

By contrast, unsupervised approaches require no labeled training data, and merely seek to discover latent structure in the data, *e.g.*, clusters [8, 16, 18, 20] or taxonomies [1, 19]. The goal in this setting is to uncover groupings of images (or image segments) that share visual patterns, with the hope that the majority of the images within a group come from the same (unfamiliar) object category.

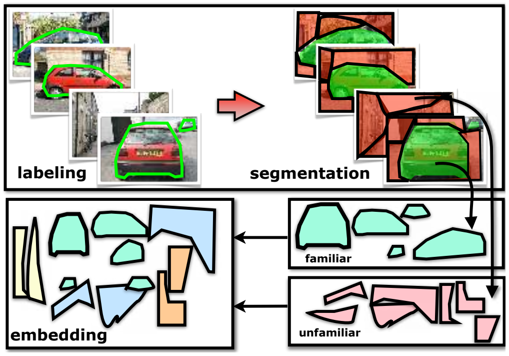

Figure 1: A set of images is partially labeled with familiar categories (*e.g.*, *car*), while unfamiliar objects are left unlabeled. Both labeled and unlabeled regions are used to learn an optimized similarity space, which facilitates discovery of unfamiliar categories in test data.

The quality of any system for unsupervised category discovery will ultimately depend upon how it determines similarity between image regions. 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 to optimize similarity for classification [5, 25]. To the best of our knowledge, there has thus far been no study of systematically optimizing a similarity function for use in unsupervised category discovery.

In this work, we propose the use of metric learning to improve the quality of unsupervised category discovery. Our framework uses an initial set of familiar categories to learn an optimal similarity space over image regions. In the optimized space, a nearest-neighbor classifier is used to determine if a new image region is familiar or unfamiliar. Then, regions predicted to be unfamiliar are collected and clustered. While our eventual goal is a full category discovery system, we focus in this work on the optimization and evaluation of the similarity space for clustering unlabeled data.

1.1. Our Approach: Improving Similarity

In our framework, we assume that a set of training images has been *partially annotated* with a set of known, *familiar categories*, so that all image regions corresponding to familiar categories have been labeled (Figure 1). All remaining, unlabeled image regions are assumed to belong to *unfamiliar categories*.

Because we do not know to *which* unfamiliar category an unlabeled training image region may belong, we cannot directly optimize a similarity function for unfamiliar categories. Instead, we train a similarity metric to discriminate between familiar categories by k-nearest-neighbor prediction. Our decision to optimize for nearest neighbor accuracy is motivated by two ideas: first, improving nearest neighbor classification provides a direct way to predict if a test segment belongs to a familiar or unfamiliar category, and second, a metric optimized to discriminate familiar categories should generalize to discriminate unfamiliar categories.

Moreover, because our framework is built upon nearest-neighbor classification, it is inherently multi-class, and automatically extends to novel classes. It can therefore be easily integrated in a continuous learning system with no need retrain each time a new category is discovered.[1] We see this as a key advantage over previous methods, where the detection of unfamiliar categories derives from the output of binary classifiers trained on familiar categories [13].

Our main technical contribution is a multiple-kernel extension to the metric learning to rank algorithm, which will allow us to learn an optimized similarity space from multiple, heterogeneous input features. Our experimental results demonstrate that learning similarity from labeled data can provide significant improvements over purely unsupervised methods. Finally, we show that including unfamiliar data during training improves the quality of the learned similarity space.

2. Discovering Object Classes

Before describing our framework in more detail, we will first introduce notation and formalize the problem.

Using ground truth label information (*e.g.*, masks or bounding boxes), each training image $\mathcal{I}$ 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. The set $\mathcal{X}_\text{m}$ contains all training segments x_i derived from ground truth annotations across all images.

Additionally, we partition each training image $\mathcal{I}$ into overlapping regions by running a segmentation algorithm multiple times. 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 $\mathcal{X}_\text{f}$. The rest of the segments, which lack (familiar) ground truth labels, are collected in the set $\mathcal{X}_\text{u}$. Throughout, we will refer to segments corresponding to familiar classes (*i.e.*, $\mathcal{X}_\text{m}$ and $\mathcal{X}_\text{f}$) as *familiar segments*, and segments corresponding to unfamiliar labels ($\mathcal{X}_\text{u}$) as *unfamiliar segments*.[2]

All segments derived from training images are collected to form the training set $\mathcal{X} = \mathcal{X}_\text{m} \cup \mathcal{X}_\text{f} \cup \mathcal{X}_\text{u}$. Although including $\mathcal{X}_\text{f}$ and $\mathcal{X}_\text{u}$ introduces some noise into the system, we demonstrate empirically in Section 4.3.1 that doing so during training improves the quality of the final similarity metric.

For each segment $x_i \in \mathcal{X}$, we compute several types of features $\phi_t(x_i)$, where each feature type ϕ_t corresponds to a space characterized by a kernel function

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

From a collection of m feature spaces over n training points, we will learn a unified similarity metric which is optimized for nearest neighbor classification.

At test time, object class discovery proceeds as follows (illustrated in Figure 2). A collection of test images $\mathcal{I}'$ are segmented multiple times to form the test set $\mathcal{X}'$. For each $x' \in \mathcal{X}'$, we use the optimized metric to locate its k-nearest neighbors from the training set, and a label for x' is predicted by the majority vote of its neighbors. Unlabeled training segments vote for a synthetic label ℓ_0, taken to mean *unfamiliar*.

After classifying each $x' \in \mathcal{X}'$, all segments with predicted label ℓ_0 are used as input to a clustering algorithm. We use spectral clustering [15] with affinities defined by a radial basis function (RBF) kernel on the learned distances:

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

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

Since our objective here is to produce a more accurate similarity space for discovery, we perform our evaluation with respect to the clustering of (predicted) unfamiliar test segments. In practice, one would follow this step by annotating the cluster with a (likely new) category label, but this step is beyond the scope of this paper.

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 Section 4.1.) In order to effectively

[1] Although one may expect to improve accuracy by re-training after the discovery of a new category, in our framework, this step is purely optional.

[2] *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.

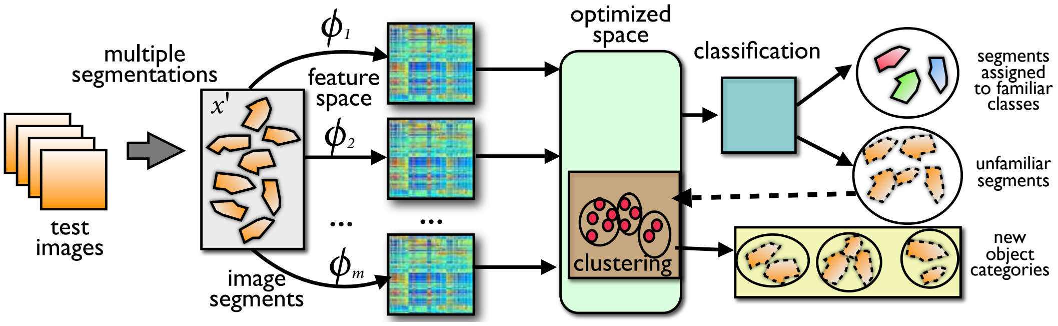

Figure 2: Discovering object classes: Each test image is partitioned into multiple segments, each of which are mapped into multiple kernel induced feature spaces, and then projected into the optimized similarity space learned by MKMLR (Algorithm 2). Each segment is classified as belonging to a familiar or unfamiliar class by k-nearest-neighbor. Unfamiliar segments are then clustered in the optimized space, enabling the discovery of new categories.

integrate heterogeneous features, we turn to multiple kernel learning (MKL) [12]. While MKL algorithms have been widely applied in computer vision applications [22, 23], most research has focused on binary classifiers (*i.e.*, support vector machines), with relatively little attention given to the optimization of nearest neighbor classifiers.

Recently, multiple kernel large margin nearest neighbor (MKLMNN) has been proposed as a method for integrating heterogeneous data in a nearest-neighbor setting [6]. Like the original LMNN algorithm [26], 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) [14], 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 1) [14] is a metric learning extension of the Structural SVM algorithm for optimizing ranking losses [10, 21]. Whereas SVM$^{\text{struct}}$ 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.

Algorithm 1 Metric Learning to Rank [14]

```
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 $C \geq 0$
Output: $d \times d$ matrix $W \succeq 0$
    $\min_{W \succeq 0,\, \xi} \quad \operatorname{tr}(W) + \frac{C}{n} \sum_{x \in \mathcal{X}} \xi_x$
    $\text{s.t.} \quad \forall x \in \mathcal{X},\ \forall y \in \mathcal{Y}:$
        $\langle W, \psi(x, y_x^*) \rangle \geq \langle W, \psi(x, y) \rangle + \Delta(y_x^*, y) - \xi_x$
```

Although ranking losses are discontinuous and non-differentiable functions over permutations, SVM$^{\text{struct}}$ and MLR resolve this issue by encoding constraints for each training point as listed in Algorithm 1. 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[3] for $x \in \mathcal{X}$, $\Delta(y_x^*, y)$ is the loss incurred for predicting y instead of y^* (*e.g.*, decrease in precision-at-k), and ξ_x is a slack variable. $\langle W, \psi(x,y) \rangle$ is the *score* function which evaluates how well the model W agrees with the input-output pair (x, y), encoded by the feature map ψ.

To encode input-output pairs, MLR uses a variant of the *partial order feature* [10] adapted for distance ranking:

$$\psi(x,y) = \sum_{i \in \mathcal{X}_x^+,\, j \in \mathcal{X}_x^-} y_{ij} \frac{D(x,i) - D(x,j)}{|\mathcal{X}_x^+| \cdot |\mathcal{X}_x^-|} \qquad (1)$$
$$D(x,i) = -(x-i)(x-i)^{\mathsf{T}}.$$

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

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

the *same class* or *different class*), and

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

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

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

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

Although Algorithm 1 lists exponentially many constraints, cutting-plane techniques can be applied to quickly find an approximate solution [11].

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.

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, *e.g.*, Tsochantaridis, et al. [21].) However, to preserve the semantics of distance ranking (Equation 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 [7, 6].

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

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

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[4] of L.

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

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

We may now formulate an optimization program similar to Algorithm 1 in terms of L:

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

The choice of $\|L\|_{\text{HS}}^2$ as a regularizer on L allows us to invoke the generalized representer theorem [17]. It follows that an optimum L^* of Equation 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:

$$\begin{aligned} \|L^*(\phi(x) - \phi(i))\|^2 &= \|M\Phi^{\mathsf{T}}\phi(x) - M\Phi^{\mathsf{T}}\phi(i)\|^2 \\ &= \|K_x - K_i\|_{M^{\mathsf{T}}M}^2 \\ &= \|K_x - K_i\|_W^2, \quad (6) \\ \text{and} \qquad \|L^*\|_{\text{HS}}^2 &= \operatorname{tr}\left(\Phi M^{\mathsf{T}} M \Phi^{\mathsf{T}}\right) \\ &= \operatorname{tr}(WK), \quad (7) \end{aligned}$$

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 Equation 5 render the program non-convex in L, which may itself be infinite-dimensional and therefore impossible to optimize directly. However, by substituting Equation 6 into Equation 4, we recover a score function of the same form as Equation 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:

$$\begin{aligned} \psi^K(x, y) &= \sum_{i \in \mathcal{X}_x^+,\, j \in \mathcal{X}_x^-} y_{ij} \frac{D^K(x,i) - D^K(x,j)}{|\mathcal{X}_x^+| \cdot |\mathcal{X}_x^-|} \quad (8) \\ D^K(x,i) &= -(K_x - K_i)(K_x - K_i)^{\mathsf{T}}. \end{aligned}$$

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

$$S_{\mathcal{H}}(L^*, x, y) = \langle W, \psi^K(x, y)\rangle. \quad (9)$$

Taken together, Equations 7 and 9 allow us to re-formulate Equation 5 in terms of W and K, and obtain a convex optimization similar to Algorithm 1. The resulting program may be seen as a special case of Algorithm 2.

[4] 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 $\|L\|_{\text{HS}} = \sqrt{\sum_i \langle v_i, v_i\rangle_{\mathcal{H}}}$.

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

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

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

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

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

$$\begin{aligned} &\min_{\{L_t\},\xi} \ \sum_{t=1}^m \|L_t\|_{\mathrm{HS}}^2 + \frac{C}{n}\sum_{x\in\mathcal{X}} \xi_x \qquad \text{s.t.} \qquad (12) \\ &\forall x, y:\ S_{\mathrm{MKL}}(\{L_t\}, x, y_x^*) \geq S_{\mathrm{MKL}}(\{L_t\}, x, y) \\ &\qquad\qquad + \Delta(y_x^*, y) - \xi_x. \end{aligned}$$

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 Equation 8. Equations 9 and 11 show that, at the optimum, S_{MKL} decomposes linearly into kernel-specific inner products:

$$S_{\mathrm{MKL}}\left(\{L_t^*\}, x, y\right) = \sum_{t=1}^m \langle W^t, \psi_t^K(x,y)\rangle. \qquad (13)$$

We thus arrive at the Multiple Kernel MLR program (MKMLR) listed as Algorithm 2. Algorithm 2 is a linear program over positive semi-definite matrices W^t and slack variables ξ, and is therefore convex.

Algorithm 2 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 $C \geq 0$
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. $\forall x \in \mathcal{X},\ \forall y \in \mathcal{Y}:$
        $\sum_{t=1}^m \langle W^t, \psi_t^K(x, y_x^*)\rangle \geq \sum_{t=1}^m \langle W^t, \psi_t^K(x,y)\rangle$
            $+ \Delta(y_x^*, y) - \xi_x$
```

We also note that like the original score function (Equation 3), S_{MKL} 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.

4. Experiments

In this section we evaluate our optimized similarity by: *(i)* the accuracy of segment classification for familiar and unfamiliar classes, *(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.

To evaluate the classification and clustering accuracy of the proposed system, we use the MSRC and PASCAL 2007 [4] databases. Our selection of these datasets was motivated by three factors:

(a) Both datasets contain at least 20 categories, multiple objects per image, and present challenges such as high intra-class, scale and viewpoint variability.

(b) 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.

(c) PASCAL presents 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. However, this makes the evaluation more realistic, as bounding boxes are a popular way of labeling objects for recognition tasks.

For experiments with MSRC, we use the same train and test split as Lee and Grauman [13] (hereafter referred to as LG10), and the object detection split of PASCAL VOC 2007 [4]. We adopt three different partitionings of each dataset into unfamiliar/familiar classes from LG10 for comparison purposes.

	Set	Unfamiliar	Familiar
	1	$1, 2, 7, 11, 20$	3–$6, 8$–$10, 12$–$19, 21$
(a)	2	1–$4, 10, 16, 17, 19$–21	5–$9, 11$–$15, 18$
	3	1–$7, 9$–$11, 13, 16$–19	$8, 12, 14, 15, 20, 21$
	1	$1, 3, 10, 14, 20$	$2, 4$–$9, 11$–$13, 15$–19
(b)	2	$1, 4$–$6, 9, 11, 14, 15, 17$–19	$2, 3, 7, 8, 10, 12, 13, 16, 20$
	3	4–$14, 16, 18$–20	1–$3, 15, 17$

Table 1: Partitions for unfamiliar and familiar classes for (a) MSRC and (b) PASCAL VOC 2007.

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

Table 2: The number of known categories ($\mathcal{L}$) and training and test segments in each partition of the datasets.

The different class partitions are shown in Table 1 and statistics of each partition are reported in Table 2.

Note that the number of examples in PASCAL VOC 07 is smaller than in MSRC. This is because PASCAL images may contain unlabeled regions, and few objects are labeled in each scene. 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.

4.1. 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 [6].

4.2. Implementation

The implementation uses the 1-slack margin-rescaling cutting plane algorithm [11] 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) [9] 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 [6].

	$\mathcal{X}_\mathrm{m}$			$\mathcal{X}_\mathrm{m} \cup \mathcal{X}_\mathrm{f}$		
Training subset	Set 1	Set 2	Set 3	Set 1	Set 2	Set 3
$\mathcal{X}_\mathrm{m}$	0.57	0.49	0.14	0.65	0.64	0.14
$\mathcal{X}_\mathrm{m} \cup \mathcal{X}_\mathrm{f}$	0.64	0.48	**0.72**	**0.68**	0.63	**0.80**
$\mathcal{X}_\mathrm{m} \cup \mathcal{X}_\mathrm{f} \cup \mathcal{X}_\mathrm{u}$	**0.65**	**0.56**	**0.72**	**0.68**	**0.66**	**0.80**

Table 3: Classification accuracy achieved for various training subsets of MSRC, and retrieval sets $\mathcal{X}_\mathrm{m}$ or $\mathcal{X}_\mathrm{m} \cup \mathcal{X}_\mathrm{f}$.

4.3. Classification accuracy

In order to evaluate the quality of our similarity space, we perform two different classification experiments: one to measure the benefits of training with unlabeled data when predicting familiar classes, and another to assess the accuracy of predicting if a test segment is familiar or not, and if so, its correct label.

4.3.1 The benefits of unlabeled data

Unlabeled data could potentially introduce noise to the metric learning step. Therefore, to objectively evaluate the contributions of labeled and unlabeled data during training, we evaluate classification accuracy by training metrics on three subsets of the training data: familiar regions ($\mathcal{X}_\mathrm{m}$), familiar regions and segments ($\mathcal{X}_\mathrm{m} \cup \mathcal{X}_\mathrm{f}$), and all training segments ($\mathcal{X}_\mathrm{m} \cup \mathcal{X}_\mathrm{f} \cup \mathcal{X}_\mathrm{u}$). Due to its dense region labeling, we focus on the MSRC dataset for this experiment. We restrict the test set to only familiar classes, and repeat the experiment for each partition of classes.

We also vary which subset of training data is used to form nearest-neighbor predictions — the *retrieval set* — at test time: either just $\mathcal{X}_\mathrm{m}$, or $\mathcal{X}_\mathrm{m} \cup \mathcal{X}_\mathrm{f}$. This allows us to evaluate the impact on accuracy due to auto-segmentation of training images.

Table 3 illustrates that including both $\mathcal{X}_\mathrm{f}$ and $\mathcal{X}_\mathrm{u}$ during training provides significant improvements in test-set accuracy. Similarly, including $\mathcal{X}_\mathrm{f}$ in the retrieval set at prediction time also provides substantial boosts in performance. This is likely due to the fact that test images are automatically segmented, and $\mathcal{X}_\mathrm{f}$ provides examples closer in distribution to the test set.

4.3.2 Classification of unfamiliar segments

We evaluate our learned similarity space by computing classification accuracy over the full test set ($\mathcal{X}'_\mathrm{f} \cup \mathcal{X}'_\mathrm{u}$). 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

	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

Table 4: Nearest-neighbor classification accuracy of MKMLR, MKLMNN, and the native feature space, including ℓ_0.

the familiar classes $\mathcal{L}$, or the *unfamiliar* class ℓ_0. The overall accuracy is reported in Table 4.

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. On 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 2).

4.4. Intra-class versus Inter-class affinities

Our second evaluation replicates an experiment on MSRC Set 1 in LG10 (Table 1, [13]). A distance matrix is computed for all pairs of test segments predicted to be unfamiliar by the segment classification step. Then, using the ground-truth labels, the average precision is computed for each test segment. Finally, the MAP score is computed for all unfamiliar classes.

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 5 shows the MAP results for both experiments. For completeness, we again compare the performance of MKMLR to MKLMNN [6]. [5]

We observe in the unbiased evaluation (Table 5b) that MKMLR outperforms the other methods under consideration for all categories.

4.5. Cluster purity

Our final 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 (Figure 5, [13]), 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 $c \in [2, 35]$, and for each c, compute the average *purity* of the clustering, where a cluster B's purity is defined as

		Airplane	Bicycle	Building	Cow	Tree
(a)	[13]	0.36	0.21	0.32	0.41	0.36
	MKLMNN	0.75	0.51	**0.38**	**0.71**	-
	Native	0.61	0.37	0.30	0.40	0.49
	Ours	**0.84**	**0.58**	**0.38**	0.41	**0.70**
(b)	MKLMNN	0.68	0.50	0.44	0.59	0.59
	Native	0.65	0.43	0.33	0.36	0.57
	Ours	**0.81**	**0.55**	**0.45**	**0.71**	**0.66**

Table 5: Comparison of MAP scores for Set 1 in MSRC. (a) MAP for segments predicted to be unfamiliar. (b) MAP on true unfamiliar segments. Test segments are correctly classified as familiar with 95% accuracy and unfamiliar with 54% accuracy.

$$\text{purity}(B) = \max_{\ell \in \mathcal{L}} |\{x' \in B \wedge \ell(x') = \ell\}| \, / |B|.$$

For each value of c, we generate 10 different clusterings, and report the average purity. The resulting mean purity curves are reported in Figure 3.

We observe that in all cases, the mean purity achieved by MKMLR is consistently above that of the native space (almost always significantly so), 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 Section 4.3.2 , the reduced size of the training set leads to some overfitting by both MKLMNN and MKMLR. However, while in Section 4.3.2 we observed a decrease in classification accuracy (compared to the native space), here we observe an *increase* in cluster purity. This indicates that MKMLR is learning some useful information which is not directly reflected in classification accuracy.

5. Conclusion

We have introduced a novel framework for improving object class discovery. By optimizing similarity by learning from a set of familiar category labels, we are able to more accurately cluster unlabeled test data. We also show that including unlabeled data during training can significantly improve the quality of the learned space. In future work, we intend to integrate this system with an active learning framework, to continuously explore large sets of object categories.

[5] In Table 5, MKLMNN has no MAP score for class *tree* because there was only one test segment of that class predicted as unfamiliar.

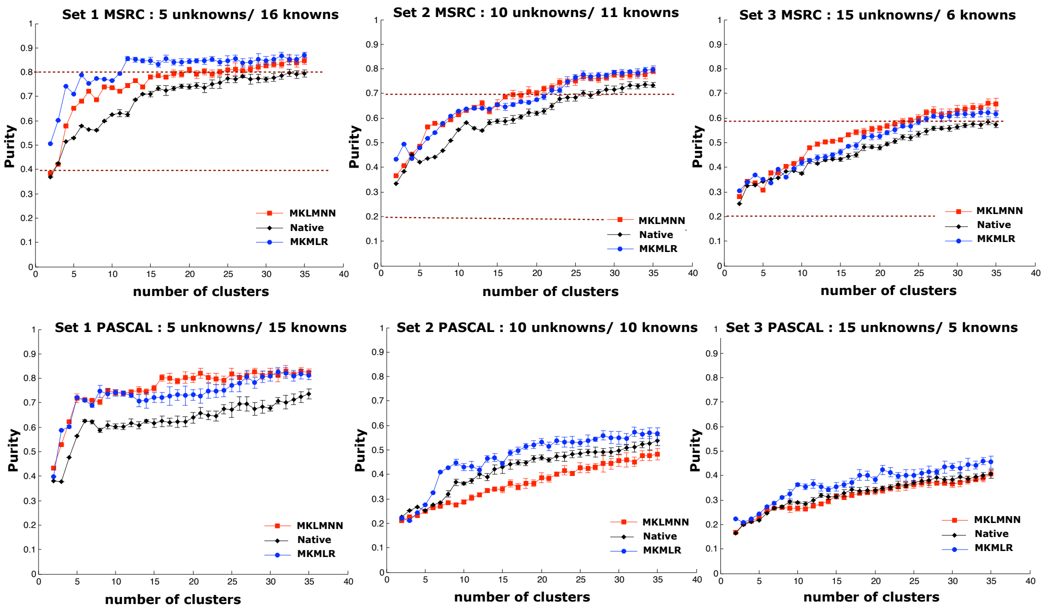

Figure 3: Mean cluster purity curves. Top plots correspond to different sets in MSRC, and bottom plots correspond to PASCAL VOC2007. Error bars correspond to one standard deviation. Dashed lines correspond to bounds on purity scores reported by LG10 (Figure 5e, [13]).

Acknowledgments: C.G. and S.B. are supported by NSF Grant AGS-0941760 and DARPA Grant NBCH1080007 subaward Z931303. B.M. and G.R.G.L. are supported by Qualcomm, Inc., eHarmony, Inc., and NSF Grant CCF-0830535. This work was supported in part by the UCSD FWGrid Project, NSF Research Infrastructure Grant EIA-0303622.

References

[1] E. Bart, I. Porteous, P. Perona, and M. Welling. Unsupervised learning of visual taxonomies. In *CVPR*, pages 1 –8, 2008.

[2] S. Branson, C. Wah, F. Schroff, B. Babenko, P. Welinder, P. Perona, and S. Belongie. Visual Recognition with Humans in the Loop. *ECCV*, pages 438–451, 2010.

[3] B. Collins, J. Deng, K. Li, and L. Fei-Fei. Towards scalable dataset construction: An active learning approach. *ECCV*, 2008.

[4] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The PASCAL Visual Object Classes Challenge 2007 (VOC2007) Results.

[5] A. Frome, Y. Singer, F. Sha, and J. Malik. Learning globally-consistent local distance functions for shape-based image retrieval and classification. In *ICCV*, pages 1–8, 2007.

[6] C. Galleguillos, B. McFee, S. Belongie, and G. Lanckriet. Multi-class object localization by combining local contextual interactions. In *CVPR*, pages 113–120, 2010.

[7] A. Globerson and S. Roweis. Visualizing pairwise similarity via semidefinite embedding. In *AISTATS*, 2007.

[8] K. Grauman and T. Darrell. Unsupervised learning of categories from sets of partially matching image features. *CVPR*, 2006.

[9] K. Järvelin and J. Kekäläinen. Cumulated gain-based evaluation of ir techniques. *ACM Trans. on Information Systems*, 2002.

[10] T. Joachims. A support vector method for multivariate performance measures. In *ICML*, pages 377–384, 2005.

[11] T. Joachims, T. Finley, and C.-N. J. Yu. Cutting-plane training of structural svms. *Mach. Learn.*, 77(1):27–59, 2009.

[12] G. R. G. Lanckriet, N. Cristianini, P. Bartlett, L. El Ghaoui, and M. I. Jordan. Learning the kernel matrix with semidefinite programming. *JMLR*, 5:27–72, 2004.

[13] Y. Lee and K. Grauman. Object-graphs for context-aware category discovery. In *CVPR*, 2010.

[14] B. McFee and G. Lanckriet. Metric learning to rank. In *ICML*, 2010.

[15] M. Meila and J. Shi. Learning Segmentation by Random Walks. *NIPS*, 2001.

[16] B. Russell, W. Freeman, A. Efros, J. Sivic, and A. Zisserman. Using multiple segmentations to discover objects and their extent in image collections. In *CVPR*, 2006.

[17] B. Schölkopf, R. Herbrich, A. J. Smola, and R. Williamson. A generalized representer theorem. In *COLT*, pages 416–426, 2001.

[18] J. Sivic, B. Russell, A. Efros, A. Zisserman, and W. Freeman. Discovering objects and their location in images. In *ICCV*, 2005.

[19] J. Sivic, B. Russell, A. Zisserman, W. Freeman, and A. Efros. Unsupervised discovery of visual object class hierarchies. In *CVPR*, pages 1 –8, 2008.

[20] S. Todorovic and N. Ahuja. Extracting subimages of an unknown category from a set of images. In *CVPR*, 2006.

[21] I. Tsochantaridis, T. Joachims, T. Hofmann, and Y. Altun. Large margin methods for structured and interdependent output variables. *J. Mach. Learn. Res.*, 6:1453–1484, 2005.

[22] M. Varma and D. Ray. Learning the discriminative power-invariance trade-off. In *ICCV*, 2007.

[23] A. Vedaldi, V. Gulshan, M. Varma, and A. Zisserman. Multiple kernels for object detection. In *ICCV*, 2009.

[24] S. Vijayanarasimhan and K. Grauman. What's it going to cost you?: Predicting effort vs. informativeness for multi-label image annotations. In *CVPR*, 2009.

[25] G. Wang, D. Hoiem, and D. Forsyth. Learning image similarity from flickr groups using stochastic intersection kernel machines. In *CVPR*, 2010.

[26] K. Q. Weinberger, J. Blitzer, and L. K. Saul. Distance metric learning for large margin nearest neighbor classification. *NIPS*, 2006.