Multi-Class Object Localization by Combining Local Contextual Interactions

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

Recent work in object localization has shown that the use of contextual cues can greatly improve accuracy over models that use appearance features alone. Although many of these models have successfully explored different types of contextual sources, they only consider one type of contextual interaction (e.g., pixel, region or object level interactions), leaving open questions about the true potential contribution of context. Furthermore, contributions across object classes and over appearance features still remain unknown.

In this work, we introduce a novel model for multi-class object localization that incorporates different levels of contextual interactions. We study contextual interactions at pixel, region and object level by using three different sources of context: semantic, boundary support and contextual neighborhoods. Our framework learns a single similarity metric from multiple kernels, combining pixel and region interactions with appearance features, and then uses a conditional random field to incorporate object level interactions. We perform experiments on two challenging image databases: MSRC and PASCAL VOC 2007. Experimental results show that our model outperforms current state-of-the-art contextual frameworks and reveals individual contributions for each contextual interaction level, as well as the importance of each type of feature in object localization.

1. Introduction

Recent work in computer vision has shown that contextual information can improve recognition of objects in real world images as it captures knowledge about the identity, location and scale of objects. Various types of contextual cues have been exploited to benefit object recognition tasks, including semantic [5, 8, 25], spatial [1, 4, 8, 11, 13, 19, 23, 27, 30, 34], scale [11, 22, 23, 31], geographic [5]. All of these models incorporate contextual information at either a global or a local image level.

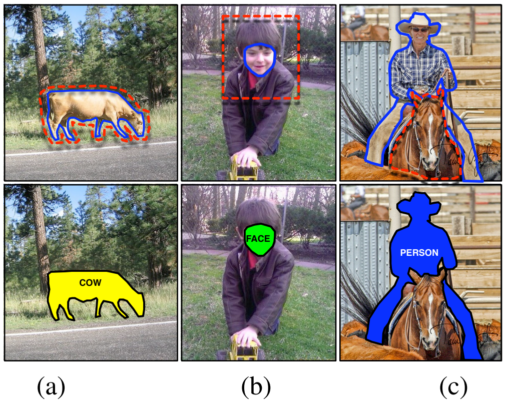


Figure 1. Examples of local contextual interactions. (a) Pixel interactions capture information such as grass and tree pixels around the cow's boundary. (b) Region interactions are represented by relations between the face and the upper region of the body. (c) Object relationships capture interactions between the objects person and horse.

Global context considers image statistics from the image as a whole scene [5, 31, 34]. Local context considers information from neighboring areas of the object, such as pixel, region, and object interactions [4, 8, 11, 22, 23, 27, 30]. *Pixel interactions* capture low-level feature interactions between spatially adjacent objects. *Region interactions* capture higher-level information from the region surrounding an object. Finally, *object interactions* capture high-level information from objects in the scene, which may be separated by large distances. Figure 1 shows examples of different contextual interaction levels. Previous models have used local context from pixels [7, 12, 14, 30], regions [1, 4, 11, 13, 17, 19], or objects [8, 23]. However, the previous models do not combine information from all *different levels* of interaction or isolate the component-wise benefits of these different levels. Therefore the relative contributions to overall performance of each interaction level remain unknown.

In this work, we present a novel framework for object localization that efficiently and effectively combines different levels of interaction. We develop a multiple kernel learning algorithm to integrate appearance features with pixel and region interaction data, resulting in a unified similarity metric which is optimized for nearest neighbor classification. Object level interactions are modeled by a conditional random field (CRF) to produce the final label prediction. Moreover, we study the relative contribution of contextual local interactions for object localization over different data sets and object classes.

2. Related Work

Many approaches for object localization use local context to improve localization accuracy [1, 7, 4, 8, 11, 13, 14, 19, 23, 30]. Although most of these models have achieved good results and some successfully combined many different sources of context at a single level, they do not combine sources from different contextual local levels or make their contributions explicit.

Previous work on image and scene classification shows that by providing a more complete representation of the scene, combining multiple contextual interaction levels can improve image classification accuracy [12, 17]. Although the explicit contributions of each level are not studied in these models, their results demonstrate the benefits of unifying contextual interactions and appearance information. However, combining these different interaction levels is a complex task, and obtaining and merging the different sources of information can be computationally expensive.

Multiple kernel learning [18] has been used in image classification [15, 32] and object localization tasks to optimally combine different types of appearance features [33] and pixel interactions [1]. These models learn convex combinations of the given base kernels, which are then used to produce classifiers, in either a hierarchical or one-versus-all framework. Although using a different similarity metric for each class has been shown to perform extremely well on these tasks [9, 32, 33], it poses a great difficulty in scaling to large datasets, and the predictions from each classifier must be combined to yield a single prediction. However, learning a single metric enables the use of nearest neighbor classification, which naturally supports multi-class problems.

3. Multi-Class Multi-Kernel Approach

In our model, each training image $\mathcal{I}$ is partitioned into segments s_i by using ground truth information. Each segment s_i corresponds to exactly one object of class $c_i \in C$, where C is the set of all object labels. These segments are collected into the training set S.

For each segment $s_i \in S$, we extract several types of features, where the pth feature space is characterized by a kernel function and inner product matrix:

$$h^p(s_i, s_j) = \langle \phi^p(s_i), \phi^p(s_j) \rangle, \quad K^p_{ij} = h^p(s_i, s_j). \quad (1)$$

From this collection of kernels, we learn a unified similarity metric over $\mathbb{R}^d$, and a corresponding embedding function $g : S \to \mathbb{R}^d$. This embedding function is used to map the training set S into the learned space, where it is then used to predict labels for unseen data with a nearest-neighbor classifier.

Because at test time, ground-truth segmentations are not available, the test image must be segmented automatically. To provide more representative examples for nearest neighbor prediction, we augment the training set S with additional segments S_A, obtained by running a segmentation algorithm multiple times on the training images [24]. Only those segments that are completely contained or overlap more than 50% with the ground truth object annotations are considered. These extra segments are then mapped into the learned space by applying $g(\cdot)$, and are also used to make label predictions on unseen data.

To counteract erroneous over-segmentation of objects, we train an SVM classifier over pairs of the extra examples S_A to predict whether two segments belong to the same object. This is then used to spatially smooth the label predictions in test images.

To incorporate context from object interactions within an image, we train a conditional random field (CRF) by using co-occurrence of objects within training images.

At test time, object localization for test images proceeds in six steps, depicted in Figure 2.

1. A test image $\mathcal{I}$ is partitioned into stable segments S'.
2. For each $s' \in S'$, we apply the learned embedding function $s' \mapsto g(s')$. (Section 3.2.)
3. The k-nearest neighbors $\mathcal{N} \subset S \cup S_A$ of $g(s')$ are used to estimate a distribution over labels $\hat{P}(C|s')$.
4. Using the pairwise SVM, the label distribution of s' may be spatially smoothed by incorporating information from other segments, resulting in a new label distribution $P(C|s')$. (Section 3.3.)
5. The conditional random field (CRF) uses object co-occurrence over the entire image to predict the final labeling of each test segment $s' \in \mathcal{I}$. (Section 3.4.)
6. Finally, to produce localizations from segment-level predictions, we consider segments to belong to the same object if they overlap and receive the same final label prediction.

3.1. Large Margin Nearest Neighbor

Our classification algorithm is based on k-nearest neighbor prediction, which naturally handles the multi-class setting. Because raw features may not adequately predict labels, we apply the Large Margin Nearest Neighbor (LMNN) algorithm to optimally distort the features for nearest neighbor prediction [35]. Neighbors are selected by using the

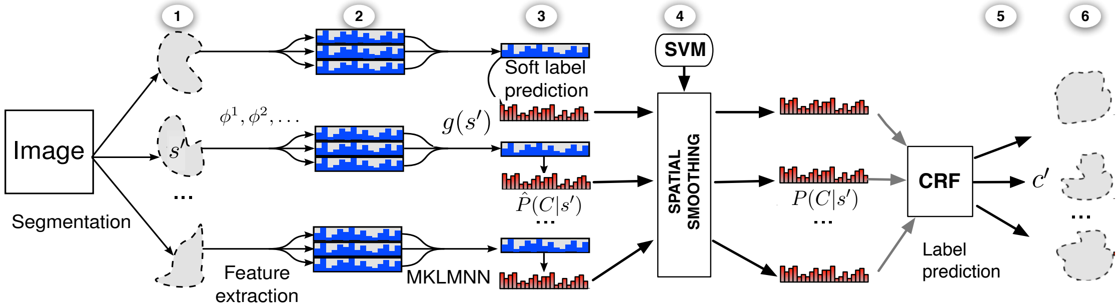


Figure 2. Our object localization framework. (1) A test image is partitioned into segments, and (2) several different features $\phi^1, \phi^2, \ldots$ (blue) are extracted for each segment. (3) Segments are mapped into a unified space by the optimized embedding $g(\cdot)$, and a soft label prediction $\hat{P}(C|s')$ (red) is computed using kNN. (4) Label predictions are spatially smoothed using a pairwise SVM, resulting in a new soft prediction $P(C|s')$. (5) A CRF estimates the final label for each test segment in the image, and (6) segments are combined into an object if they overlap and receive the same final label.

learned Mahalanobis distance metric W:

$$d(x, y) = \|x - y\|_W^2 = (x - y)^{\mathsf{T}} W (x - y) . \qquad (2)$$

Intuitively, W is trained so that for each training segment, any neighboring segments (in feature space) with differing labels are pushed away by a large margin. This is achieved by solving the following semidefinite program [3]:

$$\min_{\substack{W \succeq 0 \\ \xi_{ij\ell} \geq 0}} \sum_i \sum_{j \in \mathcal{N}_i^+} d(\phi(s_i), \phi(s_j)) + \beta \sum \xi_{ij\ell}$$
$$\forall i,\ \forall j \in \mathcal{N}_i^+,\ \forall \ell \in \mathcal{N}_i^- :$$
$$d(\phi(s_i), \phi(s_\ell)) - d(\phi(s_i), \phi(s_j)) \geq 1 - \xi_{ij\ell}, \qquad (3)$$

where $\mathcal{N}_i^+$ and $\mathcal{N}_i^-$ contain the neighbors of segment s_i in the original feature space with similar or dissimilar labels respectively, and $\beta \geq 0$ is a slack trade-off parameter. The first term in the objective minimizes the distance from each s_i to its similarly labeled neighbors s_j. The second term penalizes violations of the margin constraints, which for each s_i, force neighboring segments s_ℓ with dissimilar labels to be further away than those with similar labels (s_j).

W is a positive semidefinite (PSD) matrix which characterizes the optimal feature transformation. A linear projection matrix L can be recovered from W by its spectral decomposition, so that $W = L^{\mathsf{T}} L$:

$$W = V^{\mathsf{T}} \Lambda V = V^{\mathsf{T}} \Lambda^{\frac{1}{2}} \Lambda^{\frac{1}{2}} V \quad \Rightarrow \quad L \doteq \Lambda^{\frac{1}{2}} V. \qquad (4)$$

Here, V contains the eigenvectors of W, and Λ is a diagonal matrix containing the eigenvalues.

Although the learned projection is linear, the algorithm can be kernelized [28] to effectively learn non-linear feature transformations. After kernelizing the algorithm, it can be equivalently rewritten by representing each segment s_i by its corresponding column in the kernel matrix ($s_i \mapsto K_i$) — essentially using similarity to the training set as features — and introducing a regularization term $\gamma \cdot \mathrm{tr}(WK)$, balanced by the parameter $\gamma > 0$ to the objective function[1]. The embedding function then takes the form

$$g(s_i) \doteq LK_i. \qquad (5)$$

This embedding function generalizes to an unseen segment s' by first applying the kernel function h at s' and each s_i in the training set, and then applying the linear transformation L to the vector $(h(s', s_i))_{i=1}^n$, where $(\cdot)_{i=1}^n$ denotes vertical concatenation.

3.2. Multiple Kernel Extension

To effectively integrate different types of feature descriptions, we extend the LMNN algorithm to support multiple kernels. Previous work approaches multiple kernel learning by finding a convex combination of kernels $K^* = \sum a_p K^p$ [18].

We take a different approach here, and following [21], we learn a linear projection from each kernel's feature space where the optimization constraints are applied to the concatenated output vectors. By learning a separate projection from each feature space, we obtain a model which is more flexible than previous methods, allowing the algorithm to automatically adapt to the case where the discriminative power of a kernel varies over the data set.

To extend LMNN to the multiple kernel setting, we define the combined distance between two points by summing the distance in each (transformed) space. This is expressed algebraically as:

$$d(s_i, s_j) \doteq \sum_p \left\| K_i^p - K_j^p \right\|_{W^p}^2 . \qquad (6)$$

The regularization term $\gamma \cdot \mathrm{tr}(WK)$ is similarly extended to the sum $\gamma \cdot \sum_p \mathrm{tr}(W^p K^p)$.

Equivalently, this multiple kernel formulation can be viewed as representing each segment by concatenating its

[1] Details are omitted here for brevity. See Globerson and Roweis [10] for details of a similar construction.

columns from all kernel matrices, and learning a block-diagonal matrix where each block is a projection restricted to a particular kernel's feature space. The multiple-kernel embedding function then takes the form

$$g(s_i) = (L^p K_i^p)_{p=1}^m \,. \tag{7}$$

As in the single-kernel case, this embedding function also extends to unseen data by repeating the procedure for each kernel and concatenating the results accordingly.

We refer to this algorithm as Multiple Kernel LMNN (MKLMNN), and the optimization is listed as Algorithm 1.

Algorithm 1 Multiple Kernel LMNN (MKLMNN)

$$\begin{aligned}
&\min_{W^p,\xi} \sum_i \sum_{j \in \mathcal{N}_i^+} d(s_i, s_j) + \beta \sum \xi_{ij\ell} + \gamma \sum_{p=1}^m \operatorname{tr}(W^p K^p) \\
&\quad \forall i, \quad \forall j \in \mathcal{N}_i^+, \\
&\qquad\qquad \forall \ell \in \mathcal{N}_i^- : \quad d(s_i, s_\ell) - d(s_i, s_j) \geq 1 - \xi_{ij\ell} \\
&\qquad\qquad\qquad\qquad\quad\;\; \xi_{ij\ell} \geq 0 \\
&\quad \forall p = 1 \dots m : \quad W^p \succeq 0
\end{aligned}$$

The probability distribution over the labels for the segment s' is computed by using its k nearest neighbors $\mathcal{N} \subseteq S \cup S_A$, weighted according to distance from $g(s')$:

$$\hat{P}(C = c|s') \propto \sum_{j \in \mathcal{N}, c_j = c} \exp\left(-d(s', s_j)\right), \tag{8}$$

where c_j is the label of of segment s_j.

Although the optimization problem is convex and can be solved in polynomial time, maintaining the constraints $W^p \succeq 0$ requires a spectral decomposition and projection onto the cone of positive semidefinite matrices after each gradient step. To simplify the process, we restrict W^p to be diagonal, which can be interpreted as learning weightings over S in each feature space. The PSD projection can then be carried out by thresholding: $W_{ii}^p \mapsto \max(0, W_{ii}^p)$, and still yields good results in practice.

3.3. Spatial Smoothing by Segment Merging

Because objects may be represented by multiple segments at test time, some of those segments will contain only partial information from the object, resulting in less reliable label predictions. To counteract this effect, we smooth a segment's label distribution $\hat{P}(C|s')$ by incorporating information from segments which are likely to come from the same object, resulting in an updated label distribution $P(C|s')$.

Using the extra segments S_A automatically extracted from the training images, we train an SVM classifier to predict when two segments belong to the same object. By using the ground truth object annotations, we know when a pair of training segments came from the same object. Given two segments s_i and s_j we compute:

- pixel and region interaction features,
- overlap between segment masks,
- normalized segment centroids,
- number of segments obtained in the segmentation, and
- Euclidean distance between the two segment centroids.

We construct an undirected graph where each vertex is a segment, and edges are added between pairs that the classifier predicts should be merged. For each connected component of the graph, we merge the segments corresponding to its vertices, resulting in a new object segment s_o. We then extract features for the merged object segment s_o, apply the embedding function $g(s_o)$, and obtain a label distribution $\hat{P}(C|s_o)$ by Equation 8.

The smoothed label distribution is the geometric mean of the segment distribution and its corresponding object's distribution:

$$P(C = c|s') \propto \sqrt{\hat{P}(C = c|s') \cdot \hat{P}(C = c|s_o)}. \tag{9}$$

Note that distributions are unchanged for any segments s' which are not merged.

3.4. Contextual Conditional Random Field

Unlike pixel and region interactions, which can be described by low-level features, object interactions require a high-level description of the segment, e.g., its label. Because this information is not available until after soft label predictions are known, object interactions cannot be encoded in a base kernel. Rather, we follow the soft label prediction with a conditional random field (CRF) that encodes high-level object interactions, an approach which has been demonstrated to be effective for this task [8, 25].

In our CRF, we learn potential functions ψ from object co-occurrences, capturing long-distance dependencies between whole regions of the image and across classes. Treating the image as a bag of segments ($\mathcal{I} = \{s_i\}$), our CRF model is described as follows:

$$P\left(\vec{C} = \vec{c}|\mathcal{I}\right) \propto \exp\left(\sum_{i,j=1}^{|\mathcal{I}|} \psi(c_i, c_j)\right) \prod_{i=1}^{|\mathcal{I}|} P(C_i = c_i|s_i) \tag{10}$$

where $\vec{C} = (C_1 \dots C_{|\mathcal{I}|})$, $\vec{c} = (c_1 \dots c_{|\mathcal{I}|})$ represents the vector of labels for the segments in $\mathcal{I}$. The final label vector is the value of $\vec{c}$ which maximizes Equation 10. Since each object segment is a node in the CRF, and images contain relatively few object segments, the maximization can be carried out quickly, and the algorithm scales favorably with the number of classes. Gradient descent is used to find $\psi(\cdot)$ and Monte Carlo [26] integration to approximate the partition function.

4. Contextual Interactions

In this section, we describe the features we use to characterize each level of contextual interaction.

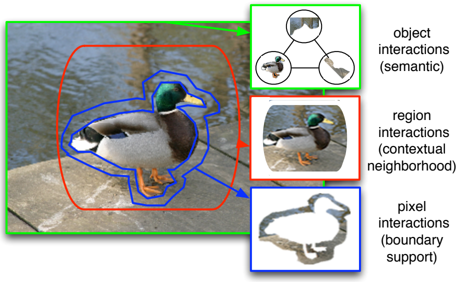


Figure 3. Local contextual interactions in our model. Pixel interactions are captured by the surrounding area of the bird. Region interactions are captured by expanding the window to include surrounding objects, such as water and road. Object interactions are captured by the co-occurrence of other objects in the scene.

4.1. Pixel Level Interactions

Pixel level interactions can implicitly capture background contextual information, because neighboring pixels tend to have similar labels (except at the discontinuities), as well as information about object boundaries. We use a new type of contextual source, *boundary support*, which corresponds to the surrounding statistics of an object in an image (as shown in Figure 3). The *boundary support* captures pixel interactions by considering individual pixel values of a surrounding region of an object.

In our model, boundary support is encoded by computing a histogram over the LAB color values between 0 and δ pixels away from the object's boundary. We compute the χ^2-distance between boundary support histograms H:

$$\chi^2(H, H') = \sum_i \frac{(H_i - H_i^{'})^2}{H_i + H_i^{'}}, \qquad (11)$$

and define the pixel interaction kernel as

$$h^{PI}(s_i, s_j; \sigma) = \exp\left(-\sigma\chi^2(H_i, H_j)\right). \qquad (12)$$

4.2. Region Level Interactions

Region level interactions have been extensively investigated in the area of context-based object localization. By using large windows around an object, known as *contextual neighborhoods* [7], regions encode probable geometrical configurations, and capture information from neighboring (parts of) objects (as shown in Figure 3). Our contextual neighborhood is computed by dilating the bounding box around the object by using a disk of diameter $d = \max\left(\sqrt{\frac{I_w}{B_w}}, \sqrt{\frac{I_h}{B_h}}\right)$, where I_w, I_h, B_w, and B_h are the widths and heights of the image and bounding box respectively. We model region interactions by computing the gist [31] of a contextual neighborhood, G_i. Hence, our region interactions are represented by the kernel

$$h^{RI}(s_i, s_j; \sigma) = \exp\left(-\sigma\chi^2(G_i, G_j)\right). \qquad (13)$$

4.3. Object Level Interactions

To train the object interaction CRF, we derive *semantic* context from the co-occurrence of objects within each training image to define the between-class potentials $\psi(c_i, c_j)$. We use simple gradient descent to find ψ that approximately optimizes the data likelihood found in the co-ocurrence matrix A. An entry $A(i, j)$ counts the times an object with label c_i appears in a training image with an object with label c_j and diagonal entries correspond to the frequency of the object in the training set.

5. Experiments

To evaluate the localization accuracy of the proposed system and study the relative importance of each interaction level, we use the MSRC and PASCAL 2007 [6] databases.

5.1. Features

Appearance Four different appearance features were computed: SIFT, Self-similarity (SSIM), LAB histogram and Pyramid of Histogram of Oriented Gradients (PHOG). SIFT descriptors [20] were computed at random locations and quantized in a vocabulary of 5000 words. SSIM descriptors [29] were computed at the same SIFT locations, and quantized in a vocabulary of 5000 words. PHOG descriptors were computed as in Bosch *et al.* [2], but only considering a 360^o orientation (608 dimensional descriptor). LAB histograms were obtained and concatenated into a 48 dimensional histogram. Finally, each feature is represented by a χ^2 kernel.

Context Two contextual features were computed using GIST (1008 dimensional descriptor) and LAB color (48 dimensional histogram). Boundary support is computed between 0 and $\delta = 20$ pixels away from the object's segment boundary.

5.2. Results

Object Localization Localization accuracy is computed by following the evaluation procedure of [6]. Table 1 shows the mean accuracy results for different combinations of appearance (App) and contextual interactions – Pixel (PI), region (RI) and object (OI) interactions – for the MSRC data set.

We observe that using only appearance information (App) results in a mean accuracy of 50%, while combining it with all local context interactions (App + PI + RI + OI) improves accuracy to 70%. Combining all local context features (PI + RI + OI) performs similarly to using appearance only, suggesting that object classes could be potentially learned by using cues that don't include appearance information [16]. If only pixel or region interactions are combined with appearance features (App+PI or App+RI), accuracy already improves over using appearance alone, where adding RI realizes a larger improvement than adding PI. Note that the object interaction model depends directly upon the estimated labels $P(C|s')$, so higher accuracy at the segment level allows the CRF to contribute better to the final localization accuracy.

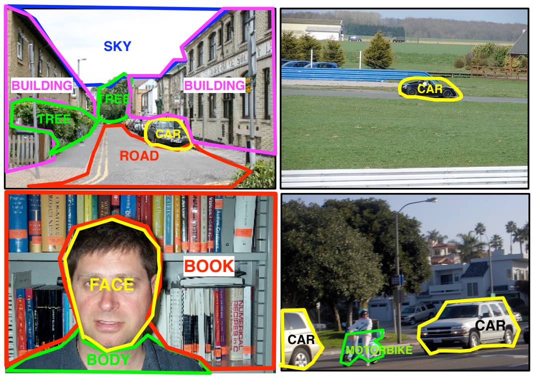
	Features	Acc	Features	Acc
MSRC	PI+RI	0.42	App+ RI	0.61
	PI+RI+OI	0.49	App+ OI	0.52
	App	0.50	App+ PI + RI	0.66
	App+ PI	0.54	App + PI + RI + OI	**0.70**
PASCAL 2007	Features	Acc	Features	Acc
	PI+RI	0.23	App+ RI	0.29
	PI+RI+OI	0.24	App+ OI	0.27
	App	0.26	App+ PI + RI	0.37
	App+ PI	0.33	App + PI + RI + OI	**0.39**

Table 1. Mean localization accuracy for the MSRC and PASCAL07 data sets. Appearance (App), pixel (PI), region (RI) and object interactions (OI) are combined for object localization.

We repeated these experiments on PASCAL07, and again evaluate the contribution of contextual interactions. Table 1 (bottom) shows the results for combing appearance with different levels of local context. As in MSRC, combining appearance with all contextual interactions (App + PI + RI +OI) improves the mean accuracy dramatically, in this case, from 26% to 39%. Pixel interactions account for the largest individual gain, improving accuracy from 26% (App) to 33% (App + PI).

However in PASCAL, adding object interactions (App + PI + RI + OI vs. App + PI + RI) improves localization accuracy by only 2%, compared to the 4% improvement in MSRC. This suggests that contextual interaction levels contribute differently to localization in the different data sets. Region interactions contribute more in MSRC by capturing probable geometrical configurations of object parts given background classes present in the scene (*sky*, *grass*, *water*, *road*, *building*). Moreover, MSRC presents more co-occurrences of object classes per image (background classes) than PASCAL, providing more information to the object interaction model. Figure 4 demonstrates these differences.

Feature Combination With respect to learning the optimal embedding, we can see in Table 2 that by using MKLMNN, we obtain substantial improvements in both data sets over both the average (unweighted combination) and single best kernels. For MSRC we achieve 66% with MKLMNN, compared to 51% with the native average kernel. Similarly, in PASCAL we observe 37% with MKLMNN, compared to 25% with the average kernel.



(a) (b)

Figure 4. (a) Examples from MSRC (left column). Region interactions capture information from neighboring (parts of) objects as background classes are often present in the scene. Furthermore, many object classes co-occur in an image, enabling object interactions to make a greater contribution to localization than in PASCAL. (b) Examples from PASCAL 07 (right column).

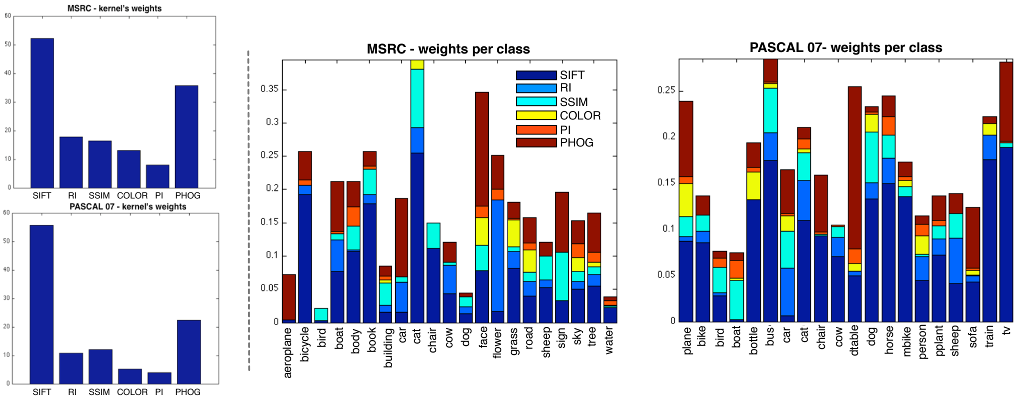
Localization	MSRC	PASCAL 07
MKL (App+PI+RI)	**0.66**	**0.37**
Average Kernel (App+PI+RI)	0.51	0.25
Best Kernel (SIFT/GIST)	0.36	0.20

Table 2. In both data sets, prediction accuracy improves significantly after learning the optimal embedding. The best accuracy using only one kernel is obtained using SIFT for MSRC and GIST for PASCAL.

In order to analyze the relative importance of each kernel in forming the optimal embedding, we examine the learned W^p matrices. As expected, the solution is sparse, since some examples are more discriminative than others for kNN classification. Figure 5(a) depicts the sum of weights assigned to each kernel. We observe that SIFT and PHOG are the most important kernels for both data sets, and color-based features receive relatively more weight in MSRC than in PASCAL. This is explained by the presence in MSRC of background classes such as *water*, *sky*, *grass* and *tree* which tend to be more homogeneous in color, therefore they can be more efficiently described using this kernel. PASCAL, on the other hand, lacks these background classes, and instead contains more "man-made objects" where color features exhibit higher variance and less discriminatory power.

Figure 5(b) illustrates the learned weighting of "neighbors" from S in each kernel, grouped by class. This demonstrates the flexibility of our multiple kernel formulation. Kernel weights automatically adapt to the regions in which they are most discriminative, as evidenced by the non-uniformity of each kernel's weight distribution. Contrast this with the more standard convex combination approach, which would assign a uniform weight to a kernel over the entire data set, potentially losing locality effects which are crucial for nearest neighbor performance.

This insight allows us to examine which features are ac-



(a)

(b)

Figure 5. Learned kernel weights for MSRC and PASCAL. (a) For kernel K^p, its total weight is $\mathrm{tr}(W^p)$. (b) Weights grouped by class.

tive for each class. For example, in MSRC, color kernels are selected for points in classes: *building*, *cat*, *face*, *grass*, *road*, *sky* and *tree*. With respect to contextual kernels, *body*, *face* and *water* give importance to pixel interactions, but not region interactions. In the particular case of class *face*, this effect is explained by the fact that faces are surrounded by (dark) hair.

Similarly, in PASCAL, classes such as *boat*, *bottle*, *chair* and *motorbike* get weights for pixel interactions and not region interaction. This is easily explained for *boats*, which are surrounded by water, for which color is highly informative. Classes *bike*, *bus*, *sheep* and *train* get weights for region interactions as they are found in proximity of other specific objects. For example, *bike* objects are often overlapped by *person* objects.

Comparison to Other Models To compare our model to current state-of-the-art algorithms, we compute mean localization accuracy per class. Table 3 shows mean accuracy for our model and other models for MSRC. We outperform [8] in half of the classes, and obtain higher average accuracy overall, demonstrating the benefit of combining different contextual interaction levels. We compare our model to the state-of-the-art model [33] and the best results for localization in PASCAL 07 challenge [6], as well as two contextual models [1, 8]. Table 4 shows the localization accuracy for each class. We notice that our model performs best in the largest number of classes (tied with [8]), and we achieve a higher mean localization accuracy.

Our multiple kernel framework for learning a single metric over all classes outperforms models which learn class-specific kernel combinations [1, 33]. This owes to the fact that our embedding algorithm is geared directly toward multi-class prediction, and information can be shared between all classes by the joint optimization. Moreover, models in [1, 33] report only modest gains over the unweighted average of base kernels, but our model achieves significant improvement over both the average and best kernels. This suggests that convex combinations of kernels may be too restrictive, but our approach of concatenated linear projections provides a greater degree of flexibility to the model.

Implementation Details The same data split of Galleguillos *et al.* [8] was used for MSRC and [6] for PASCAL, where only 30 images per class where used for training our model. Multiple stable segmentations were computed with 9 different segmentations (from 2 to 10) which together results in 54 segments per image. We train our MKLMNN algorithm using 15 neighbors, and parameters β and γ are found using cross-validation. For MSRC, the results are stable for a wide range of k values, between 5 and 15, and k = 10 is used, and for PASCAL we choose the best k by using the validation set. χ^2 kernels ($\sigma = 3$) were used for all training and test kernel matrices. For our CRF, we the use same parameters as [8].

6. Conclusion

In this paper, we have introduced a novel framework that efficiently and effectively combines different levels of local context interactions. Our multiple kernel learning algorithm integrates appearance features with pixel and region interaction data, resulting in a unified similarity metric which is optimized for nearest neighbor classification. Object level interactions are modeled by a conditional random field (CRF) to produce the final label prediction. We examined the contribution of each contextual interaction and by combining these levels we obtain significant improvement over current state-of-the-art contextual frameworks. We believe that by adding another object interaction type, such as spatial context [8], localization accuracy could be improved further.

Acknowledgements: C.G. and S.B. are supported by NSF Grant AGS-0941760 and DARPA Grant NBCH1080007

	aeroplane	bike	bird	boat	body	book	building	car	cat	chair	cow	dog	face	flower	grass	road	sheep	sign	sky	tree	water	mean
App	0.49	0.95	0.00	0.31	0.35	0.38	0.65	0.51	0.09	0.66	0.45	0.13	0.40	0.33	0.93	0.62	0.55	0.63	0.53	0.91	0.54	0.50
App+PI+RI	0.96	1.00	0.10	0.63	0.66	0.74	0.63	0.86	0.18	0.69	0.76	0.27	0.60	0.72	0.94	0.71	0.95	0.70	0.47	0.70	0.50	0.66
Ours	**1.00**	**0.98**	0.11	0.63	0.55	**0.78**	0.73	**0.88**	0.11	**0.80**	**0.74**	0.43	0.72	**0.72**	**0.96**	0.76	**0.90**	**0.92**	0.50	0.76	0.61	**0.70**
[8]	0.73	0.60	**0.52**	**0.81**	**0.77**	0.56	**0.91**	0.57	**0.42**	0.37	0.41	**0.46**	**0.81**	0.65	0.95	**0.96**	0.55	0.54	**0.97**	**0.80**	**0.95**	0.68

Table 3. First two rows: localization accuracy for our system using appearance alone (App) and together with pixel and region interactions (App+PI+RI). Last two rows: Comparison of localization accuracy between different systems for MSRC object classes. Results in bold indicate the best performance per class. Our system achieves the best average accuracy.

	plane	bike	bird	boat	bottle	bus	car	cat	chair	cow	dtable	dog	horse	mbike	person	pplant	sheep	sofa	train	tv	mean
Ours	0.33	0.24	**0.47**	**0.69**	0.22	0.37	**0.71**	0.33	0.07	0.15	**0.74**	0.21	0.26	**0.55**	0.33	**0.29**	0.38	0.23	**0.51**	0.57	**0.39**
[33]	0.38	**0.48**	0.15	0.15	0.22	**0.51**	0.51	0.30	0.17	**0.33**	0.23	0.22	**0.51**	0.46	0.23	0.12	0.24	0.29	0.45	0.49	0.32
[8]	**0.63**	0.22	0.14	0.42	**0.43**	0.50	0.62	0.32	**0.37**	0.19	0.30	0.29	0.15	0.31	**0.43**	0.33	**0.41**	**0.37**	0.29	**0.62**	0.37
[1]	0.11	0.12	0.09	0.06	0.00	0.25	0.14	**0.36**	0.09	0.14	0.24	**0.32**	0.27	0.34	0.03	0.02	0.09	0.30	0.30	0.08	0.17
[6]	0.26	0.41	0.10	0.09	0.21	0.39	0.43	0.24	0.13	0.14	0.10	0.16	0.34	0.38	0.22	0.12	0.18	0.15	0.33	0.29	-

Table 4. Comparison of localization accuracy between different systems for PASCAL 07 object classes. Results in bold indicate the best performance per class. Our system achieves the best average accuracy.

subaward Z931303. B.M. and G.R.G.L. are supported by NSF Grant DMS-MSPA 0625409 and eHarmony, Inc.

References

[1] M. Blaschko and C. H. Lampert. Object localization with global and local context kernels. In *BMVC*, 2009.

[2] A. Bosch, A. Zisserman, and X. Munoz. Scene classification via pLSA. *ECCV*, 2006.

[3] S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.

[4] C. Desai, D. Ramanan, and C. Fowlkes. Discriminative models for multi-class object layout. *ICCV*, 2009.

[5] S. K. Divvala, D. Hoiem, J. H. Hays, A. A. Efros, and M. Hebert. An empirical study of context in object detection. In *CVPR*, June 2009.

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

[7] M. Fink and P. Perona. Mutual boosting for contextual inference. *NIPS*, 2004.

[8] C. Galleguillos, A. Rabinovich, and S. Belongie. Object categorization using co-occurrence, location and appearance. *CVPR*, 2008.

[9] P. Gehler and S. Nowozin. On Feature Combination for Multiclass Object Classification. *ICCV*, 2009.

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

[11] S. Gould, R. Fulton, and D. Koller. Decomposing a Scene into Geometric and Semantically Consistent Regions. *ICCV*, 2009.

[12] X. He, R. S. Zemel, and M. Á. Carreira-Perpiñán. Multiscale conditional random fields for image labeling. In *CVPR*, 2004.

[13] G. Heitz and D. Koller. Learning spatial context: Using stuff to find things. In *ECCV*, volume 1. Springer, 2008.

[14] H. Kruppa and B. Schiele. Using local context to improve face detection. *BMVC*, pages 3–12, 2003.

[15] A. Kumar and C. Sminchisescu. Support kernel machines for object recognition. *ICCV*, 2007.

[16] N. Kumar, A. Berg, P. Belhumeur, and S. Nayar. Attribute and Simile Classifiers for Face Verification. *ICCV*, 2009.

[17] S. Kumar and M. Hebert. A hierarchical field framework for unified context-based classification. In *ICCV*, 2005.

[18] 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.

[19] J. J. Lim, P. Arbelaez, C. Gu, and J. Malik. Context by region ancestry. *ICCV*, 2009.

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

[21] B. McFee and G. R. G. Lanckriet. Partial order embedding with multiple kernels. In *ICML*, 2009.

[22] K. Murphy, A. Torralba, and W. Freeman. Using the forest to see the trees: a graphical model relating features, objects and scenes. *NIPS*, 16, 2003.

[23] D. Parikh, C. Zitnick, and T. Chen. From Appearance to Context-Based Recognition: Dense Labeling in Small Images. In *CVPR*, 2008.

[24] A. Rabinovich, T. Lange, J. Buhmann, and S. Belongie. Model order selection and cue combination for image segmentation. In *CVPR*, 2006.

[25] A. Rabinovich, A. Vedaldi, C. Galleguillos, E. Wiewiora, and S.Belongie. Objects in context. *ICCV*, 2007.

[26] C. P. Robert and G. Casella. *Monte Carlo Statistical Methods*. Springer-Verlag New York, Inc., 2005.

[27] B. Russell, A. Torralba, C. Liu, R. Fergus, and W. Freeman. Object Recognition by Scene Alignment. *NIPS*, 2007.

[28] B. Scholkopf and A. J. Smola. *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. MIT Press, 2001.

[29] E. Shechtman and M. Irani. Matching local self-similarities across images and videos. In *CVPR*, 2007.

[30] J. Shotton, M. Johnson, and R. Cipolla. Semantic texton forests for image categorization and segmentation. In *CVPR*, 2008.

[31] A. Torralba. Contextual priming for object detection. *IJCV*, 2003.

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

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

[34] J. Verbeek and B. Triggs. Scene segmentation with crfs learned from partially labeled images. *NIPS*, 2008.

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