Contextual Object Detection Using Set-Based Classification

Ramazan Gokberk Cinbis[1] and Stan Sclaroff[2]

[1] LEAR, INRIA Grenoble, France
[2] Department of Computer Science, Boston University, USA

Abstract. We propose a new model for object detection that is based on set representations of the contextual elements. In this formulation, relative spatial locations and relative scores between pairs of detections are considered as sets of unordered items. Directly training classification models on sets of unordered items, where each set can have varying cardinality can be difficult. In order to overcome this problem, we propose SetBoost, a discriminative learning algorithm for building set classifiers. The SetBoost classifiers are trained to rescore detected objects based on object-object and object-scene context. Our method is able to discover composite relationships, as well as intra-class and inter-class spatial relationships between objects. The experimental evidence shows that our set-based formulation performs comparable to or better than existing contextual methods on the SUN and the VOC 2007 benchmark datasets.

1 Introduction

Object detection is among the most difficult problems in computer vision. Part of this difficulty is caused by ambiguities in low-level features due to background clutter, occlusions, etc., and also by the large amount of variance in the domain. Recent studies [1,2,3] have shown that modeling contextual relationships can help overcome such challenges and improve object detection performance.

In many computer vision tasks, image entities can naturally be represented as sets of items, where each item is a high-dimensional data point. For example, an object in an image can be represented by a set of local image patches. These image patches can be described with various image statistics, such as color and/or shape features. Similarly, the scene of an image can be represented by the set of detected objects within.

Based on these observations, we argue that the contextual relationships can also be represented in terms of sets, where each contextual element is a high-dimensional item in the set. In this representation, two main contextual relationships can be considered. First is the *object context*: some object classes tend to co-occur in particular spatial arrangements, e.g., a person riding a horse. Second is the *scene context*: some objects tend to be in particular arrangements with respect to the overall scene layout, e.g., cars in a street.

In order to model object context, we first use single object detectors to obtain object detection candidates. We then consider each detection as a reference object and represent its context via the set of the other candidate detections in the

A. Fitzgibbon et al. (Eds.): ECCV 2012, Part VI, LNCS 7577, pp. 43–57, 2012.

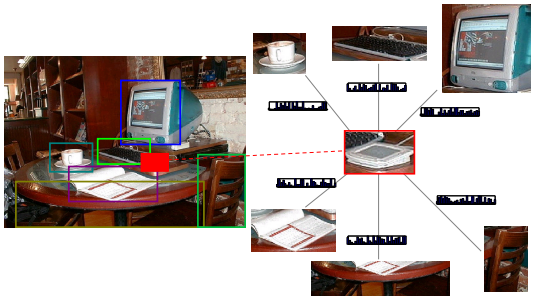


Fig. 1. Illustration of our set-based contextual object detection approach. The example reference object(touchpad) is shown with a red rectangle. The contextual properties of each detection relative to the reference object is encoded by a feature vector. The set of all descriptors define the contextual representation surrounding the reference object. The process is repeated by considering each candidate detection as the reference object.

image. Fig. 1 illustrates our representation. The reference object (touchpad) can be difficult to recognize by itself due to ambiguities in the local features or unfamiliar appearance. However, the other objects in the image can give strong cues that the reference object may be a touchpad. To account for this, our context model includes the set of descriptors for the other objects detected in the scene. We describe each item using its detection score, class, and relative bounding box with respect to the reference object. By evaluating these object detections in a set framework, we rescore each of the detections based on the other candidate detections. In addition to object context, we also model scene context in terms of coarse scene shape and the object's spatial position.

Representations that are based on sets of (unordered) high-dimensional items, where each set can have varying cardinality, can be difficult to use directly. Therefore, many of the existing approaches use either individual items, or intermediate representations. Individual items may not contain sufficient information for accurate classification. Using intermediate representations, such as a histogram of quantized items [4], is a common trick to build models. However, such intermediate representations may not be optimum for the final classification task.

In order to work directly with sets and bypass the intermediate representations, we propose a discriminative learning algorithm for set classification, which we call SetBoost. SetBoost is a weakly-supervised learning algorithm in the sense that it requires labels of sets during training but it does not require item labels. This feature allows the algorithm to deal with irrelevant and noisy items.

Our context model has several notable features. First, it can learn the contextual relationships of objects without predefining the relationship types. For example, it can discover composite relationships like "mouse above a table and below a screen", without the need for explicit categorization of spatial relations. Second, it can learn both intra-class and inter-class spatial relationships. Third, our formulation can learn to select one or more detections from a set of overlapping detections and implicitly execute non-maxima suppression[2].

To evaluate the performance of our approach, we use two object detection benchmark datasets: VOC 2007 [5] and SUN [3]. Average precision (AP) scores [5] are used to quantify performance. When we apply our context model to the outputs of the baseline object detectors, we observe significant improvements both on the VOC 2007 dataset and on the SUN dataset. By these results, we demonstrate that our set-based contextual representation provides an improvement over state-of-the-art object context models [3,2] and performs comparable to [6], which combines background context and object context. We also show that the proposed SetBoost algorithm is more effective than using bag-of-words or PMK [7].

2 Related Work

Contextual relationships can be represented in terms of the relations between local image patches [8,9], objects [10,2,3], coarse scene characteristics [11], local background regions [6] and the geometric layout of the scene [12] (see [13] for a review). In this paper, we focus on modeling the relationships of objects with respect to other objects and the global scene characteristics.

Choi et al. [3] propose a tree-structured graphical model to encode object and scene context. Each detection is mapped to a 3D coordinate frame according to predefined object heights, assuming that the camera type and object heights do not change drastically. Spatial relationships are modeled by fitting Gaussian distributions to the relative 3D positions of the object class pairs. In contrast to the generative training of [3], our context model is discriminatively trained. In addition, we do not make assumptions about the underlying distribution of spatial relationships of objects.

Desai et al. [2] use a structured prediction model to encode relationships of pairs of objects. They quantize the relative locations and sizes of object bounding boxes into predefined categories (above, below, etc.) and the weights of these relationship categories are learned via Structural SVMs. We do not predefine relationship categories; instead, we learn relationships that are important for contextual object detection. Moreover, we directly train our discriminative model to "rescore detections", whereas [2] first trains the model to select a subset of detections and then uses an approximation to rescore these detections.

In [14], each detection is re-scored according to its location and the score of the top detection from each class. In contrast, our approach utilizes multiple detections from each class and their relative spatial relationships, which allows much richer context models.

Set-Based Classification. One way to formulate the set-based learning problem is to use *set kernels*. Some kernels proposed for this purpose compare parametric distributions of items [15,16], some find correspondences between the items explicitly [17] or implicitly [7]. The Pyramid Match Kernel (PMK) [7] is particularly appealing as an efficient and effective set kernel. For a given pair of sets, PMK first quantizes the feature space into a multi-resolution grid and counts the number of items for each set within each grid cell. SetBoost differs

from PMK in the sense that it can be used to discriminatively learn different weights for different feature dimensions. For example, in SetBoost with decision trees, the partitions and their weights are found by discriminative training, whereas in PMK, they are defined by a heuristic. The learned partitioning and weights in SetBoost can provide appropriate feature selection.

In contrast with codebook based approaches [18,19,4], rather than optimizing an intermediate codebook, SetBoost directly minimizes the loss function on the training data. In this respect, our approach bears similarities to [20], which includes a boosting algorithm using sets of interest points for image classification. However, [20] supports only a specific loss function and SVM item classifiers, whereas SetBoost can be used with any non-increasing margin loss function and any item classifier. Moreover, spatial information is ignored in [20].

Multiple Instance Learning (MIL) has some similarities to set-based classification. For example, in [21], AdaBoost is used with a (weak) MIL classifier for object detection. In MIL, the aim is commonly to learn a model for classifying individual instances, instead of bags. This differs from the set-based classification problem, where the items in a set together form a descriptor.

3 Contextual Object Detection

Contextual object detection tries to exploit the fact that many objects tend to exist in particular arrangements in natural scenes. Object-object and object-scene relationships are two important types of contextual cues. Object-object relationships include co-occurrences, relative positions, and relative sizes of objects, e.g., "keyboard and mouse". Object-scene relationships include expected object presence and position according to the characteristics of the scene, e.g., "refrigerator in a kitchen". In this section, we first review how we represent the object-object and object-scene context in terms of sets. In the next section, we present our algorithm for learning this set-based context model.

3.1 Modeling Object-Object Relationships

To model object-object relationships that may be present within images, we first apply available object detectors and obtain a set of detections together with their initial classification scores. We then use our context model to rescore each detection based on all other detections in the image, i.e., we find the probability of a detection given the evidence from all other detections in the image.

A set-based representation follows our definition of this contextual rescoring. We consider each detection as the *reference object* and the rest of the detections in the image as items of the set representing the object-object context for the reference object. Each item is represented by a feature vector that encodes the spatial relationship, class and detection confidence of the item with respect to the reference object. A classifier is then used to rescore the reference object detection based on this set of feature vectors computed for the other detections.

More formally, given the set of object detections in an image, each detected object d has a score d_s normalized to the range $[\epsilon, 1]$ where $\epsilon > 0$ ($\epsilon = 0.01$ in

Table 1. Features used in our set-based representation. Each detection is represented by a small number of features computed relative to the reference object.

Feature	Value	Feature	Value
classwise encoded score	$[0, ..., 0, d_s, 0, .., 0]$	distance	$\left\|(d_y - d_y^{\text{ref}})^2 + (d_x - d_x^{\text{ref}})^2\right\| / d_h^{\text{ref}}$
relative y	$(d_y - d_y^{\text{ref}})/d_h^{\text{ref}}$	overlap	$\left\|d_B \cap d_B^{\text{ref}}\right\| / \left\|d_B \cup d_B^{\text{ref}}\right\|$
relative y (abs)	$\|d_y - d_y^{\text{ref}}\|/d_h^{\text{ref}}$	score ratio	d_s/d_s^{ref}
relative height, width	d_h/d_h^{ref}, d_w/d_w^{ref}	score difference	$d_s - d_s^{\text{ref}}$

our experiments), a class $d_c \in \{1, 2, ..., C\}$, where C is the number of classes, and a bounding box $d_B = [d_x, d_y, d_h, d_w]$ (x, y location, height and width). We consider each detected object as the reference object d^{ref} and update its detection score according to its contextual relationships with the other detections in the image. Context for the reference object is represented as a set X. Each item $\mathbf{x} \in X$ is the feature vector for each other detection $d \neq d^{\text{ref}}$. The formulae for the features used in forming $\mathbf{x}$ are given in Table 1 and summarized below.

Classwise encoded score: We create a vector of length C where its c-th dimension is set to d_s and all other dimensions are set to zero. In this encoding, a single threshold on a score dimension (e.g., decision tree node) can filter both the class and score of a detection.

Spatial relations: Several features encode spatial relations of objects. We normalize these features with respect to the height (or width) of the reference object's bounding box to achieve a degree of invariance to scale changes.

- **Relative y**: Many object class pairs have a distinctive relative y location, e.g., bicycle-person. We do not consider the relative x location since it varies significantly due to changes in perspective.
- **Relative y (abs):** This feature is useful in cases where the relative vertical location can be "above" or "below". For example, objects on a table may appear above or below the center of the table's bounding box.
- **Relative height and width:** We introduce these features to encode the relative scales of objects.
- **Distance:** Contextual interaction may decrease as objects appear farther away from each other.

Overlap ratio: A high overlap between a pair of objects may indicate high contextual coherence, as well as, conflict. For example, an overlapping "mug and table" pair is likely to be contextually coherent, whereas one of the detections in an overlapping "car and table" pair is likely to be incorrect.

Relative scores: These features are intended to help in learning inter-class and intra-class relationships. If one of the object detections has a "significantly" higher score, then the context model may prefer that detection. The "significance" depends on the object classes, and can be learned during training.

In the end, each item descriptor $\mathbf{x}$ is a $C + 8$ dimensional vector, where C is the number of object classes. We evaluate the effect of these features in Sec. 5.

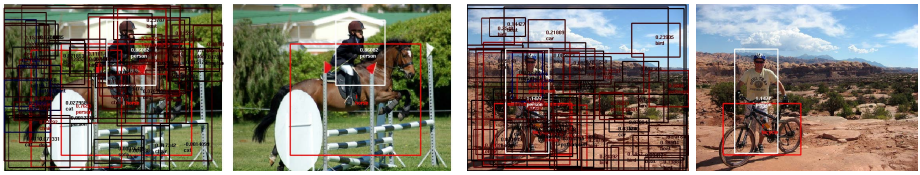


Fig. 2. Example learned relationships in the VOC 2007 dataset. White bounding boxes show the reference object with its final score. Red (blue) bounding boxes represent positively (negatively) voting detections. The images to the left show all objects with their contextual votes, where the intensity of colors is scaled with respect to magnitude of the vote. Many noisy/irrelevant detections have approximately zero vote. The images to the right show a reference detection and the detection with the highest contextual influence.

Learning. During training, we assume that the training images have ground truth annotations of object bounding boxes for the classes of interest. We first run the baseline object detectors for each class. Then, we evaluate each output of the object detectors using the ground truth bounding boxes and the VOC detection evaluation criteria [5]. The true positive detections are used as the positive training samples, whereas the false positives are used as the negative training samples for our algorithm. Given the resulting training set, we learn an object-object context model for each object class separately via our learning framework as described in Sec. 4.

In Fig. 2, we show two examples of relationships learned by our model on the VOC 2007 dataset [5]. In each example, a person detection is used as the example reference object (shown with a white bounding box). The other detected objects are colored by their estimated degree of contextual interaction with the reference object, where red and blue colors correspond to positive and negative interactions, respectively. We observe that our model estimates strong coherence between the pairs "person-horse" and "person-bicycle" in these examples.

3.2 Modeling Object-Scene Relationships

For modeling relationships between objects and the scene (*scene context*), we use GIST descriptors to describe global scene characteristics [11] and encode the relationships between the objects and the scene as a feature vector. For each detection d in an image with height I_h and width I_w, we create the feature vector $\mathbf{u}_d = [\frac{d_y}{I_h}, \frac{d_h}{I_h}, \frac{d_w}{I_w}, \text{GIST}]$.

Our scene context model has similarities to[3], which also uses GIST descriptors. However, we learn a classification model over spatial position, i.e., the location and size of objects and the scene characteristics jointly, whereas spatial information is discarded in the scene context model of [3].

Learning. Any vector classifier might be employed to classify a given $\mathbf{u}_d$ vector. In our framework, we use SetBoost, by considering each $\mathbf{u}_d$ as a singleton set, with decision tree weak classifiers. For training, we use the same approach that

we use in the object-object context model, i.e. for each class we train a classifier over the true and false positive detections. In a test image, a scene context score is obtained by applying the corresponding model to each detection.

3.3 Combining Scores

The object and scene context models provide separate scores for each (reference) object. These models do not utilize the detection scores of the reference objects directly; therefore, it is important to combine the raw detection score d_s with the context-based scores for the final classification. For this reason, we always use a linear combination of the scores given by the context model(s) and the baseline detector. Combination weights are determined via linear SVMs training.

4 Learning Set Classifiers via SetBoost

Given training data, we want to build a classifier that operates on sets. Let $\{X_1, ..., X_N\}$ be the training examples. Each training example X_i (i.e., context for i-th reference object) is a set of high-dimensional points $\mathbf{x} \in X_i$, referred to as *items* (i.e., context descriptors). For the sake of brevity, we assume that each item is a d-dimensional vector. Each X_i has an associated binary class label $y_i \in \{-1, +1\}$ corresponding to true positive or false positive detection.

Our goal is to learn a classification function $G : \mathcal{X} \rightarrow \mathbb{R}$ such that it minimizes the total loss $\mathcal{L}(G) = \sum_{i=1}^{N} L(y_i G(X_i))$ where $L(z) : \mathbb{R} \rightarrow \mathbb{R}$ is a loss function (We use exponential loss $L(z) = e^{-z}$) and $\mathcal{L}$ is the loss functional. In accordance with the boosting terminology, we define *strong set classifier* $G(X) = \sum_{t=1}^{T} F_t(X)$, where each $F_t : \mathcal{X} \rightarrow \mathbb{R}$ is a *weak set classifier*.

Each weak set classifier F_t is based on a *weak item classifier* f_t:$\mathbb{R}^d \rightarrow \mathbb{R}$ weighted by some non-negative α_t:

$$F_t(X) = \alpha_t \sum_{\mathbf{x} \in X} k_x f_t(\mathbf{x}) \tag{1}$$

where the k_x are (optional) item weights. Defining k_x allows us to introduce prior knowledge on the relevance of an item. We set the k_x to the item detection scores in our context model to emphasize high-confidence detections.

We use functional gradient descent interpretation of boosting, specifically MarginBoost framework [22], in developing our learning algorithm. According to this interpretation, at each iteration, a new weak classifier F is added such that F minimizes the loss functional $\mathcal{L}(G + \epsilon F)$. Assuming L is a non-increasing function, $\mathcal{L}(G + \epsilon F)$ can be minimized by choosing the weak classifier that maximizes

$$-\left\langle \nabla \mathcal{L}\left(G\right), F \right\rangle \propto -\sum_{i=1}^{N} y_i F(X_i) L'(y_i G(X_i))\,, \tag{2}$$

where $L'(z)$ is the derivative with respect to z and $\nabla \mathcal{L}\left(\mathcal{G}\right)$ is the gradient with respect to G. Substituting Eq. 1 into Eq. 2, we obtain the objective function

$$-\alpha \sum_{i=1}^{N} \sum_{\mathbf{x} \in X_i} y_i k_x f(\mathbf{x}) L'(y_i G(X_i))\,.$$

Since $f(\mathbf{x})$ is a binary classifier, i.e., $f : \mathbb{R}^d \to \{-1, +1\}$, it is equivalent to train f by minimizing the following proxy loss, which is defined in terms of individual items:

$$\sum_{i=1}^{N} \sum_{x \in X_i} D(i) k_x \left[f(\mathbf{x}) \neq y_i \right] , \tag{3}$$

where $D(i)$ is the *sample weight* of the i^{th} set,

$$D(i) = \frac{L'(y_i G(X_i))}{\sum_{i=1}^{N} L'(y_i G(X_i))} . \tag{4}$$

At each iteration, we train an item classifier according to the weights D. In the SetBoost reweighting mechanism, D is updated in order to learn item classifiers that serve as a good discriminant on the sets instead of individual items. Therefore, the set classifier learning problem is reduced to a series of intermediate item classification problems.

At iteration t, the weight α_t of the new weak set classifier is found by minimizing the total loss $\sum_{i=1}^{N} L\left(y_i \left(G(X_i) + \alpha F(X_i)\right)\right)$ using numerical optimization. We use the LBFGS-B algorithm [23].

SetBoost can also be considered as a generalization of AdaBoost [24] since SetBoost simplifies to AdaBoost when each observed set comprises a single item.

4.1 Effective Utilization of Decision Trees

One suitable weak item classifier choice is decision trees. At every iteration of SetBoost, a decision tree minimizing Eq. 3 will be learned. A tree with M leaves will partition the feature space into partitions $P_1, ..., P_M$. We can assign weights α^m to decision tree leaf nodes based on their discriminative power for set classification.

Let the $H^m(X) = \sum_{\mathbf{x} \in X, \mathbf{x} \in P_m} k_x$, which is the total prior vote weight of items that are grouped into the partition P_m for a training example X. Once a decision tree is built, we choose leaf node weights by minimizing the total loss:

$$\alpha = \underset{\alpha^1, ... \alpha^M}{\arg\min}\ \lambda \alpha^{\mathrm{T}} \alpha + \sum_{i=1}^{N} L\left(y_i G(X_i) + y_i \sum_{m=1}^{M} \alpha^m H^m(X_i) \right) . \tag{5}$$

We use LBFGS-B [23] to optimize over the α values. Optimizing the α values requires only a modest fraction of the time needed to build decision trees.

4.2 Stochastic Training

The training time can be an issue for datasets that contain high cardinality sets. To address this, we propose a sampling-based implementation, summarized in Alg. 1. At each training iteration, a fixed number of sets is sampled. Training examples are subsampled with respect to the sample weight distribution D, rather than from a uniform distribution as in [25] , in order to "catch" more

Algorithm 1. Stochastic SetBoost with decision trees. T is the number of weak classifiers used in constructing $G(X)$, N is the total number of examples in the training set, I is the set of sampled example indices at an iteration, where $|I| \leq N$.

1. Initialize $D(i) = 1/N$, $i = 1...N$ and $G(X) = 0$.
2. For $t = 1$ to T
   (a) Sample subset of indices I with respect to D.
   (b) Train $f_t(\mathbf{x})$ to min. $\sum_{i \in I} \sum_{\mathbf{x} \in X_i} D(i) \left[f_t(\mathbf{x}) \neq y_i \right]$
   (c) Solve for $[\alpha_t^1, ..., \alpha_t^M]$ via Eq. 5
   (d) Let $F_t(X) = \sum_{m=1}^{M} \alpha_t^m H_t^m(X)$
   (e) $D(i) \leftarrow D(i) \exp\left(-y_i F_t(X_i)\right)$, $i = 1...N$
   (f) Normalize D to a probability distribution.
3. Obtain $G(X) = \sum_{t=1}^{T} F_t(X)$

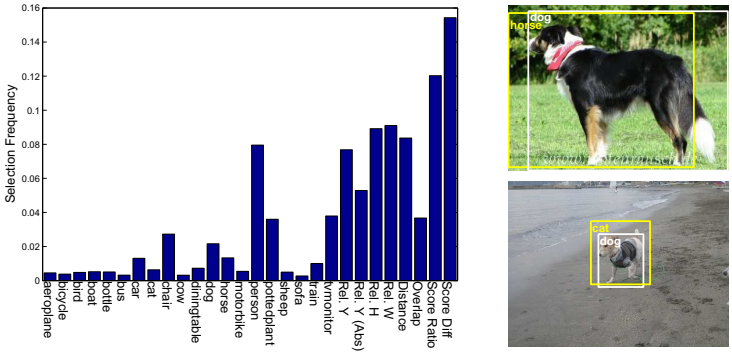
difficult examples at each iteration. We set the sample size $|I|$ to the average number of positive training examples among classes in a dataset. As shown in Alg. 1, $D(i)$ is still utilized after the subsampling step.

Another potential problem is training with large class imbalances. The number of negative training examples is often much larger than the number of positive training examples. To address this problem, we sample an equal number of positive and negative examples at each boosting iteration. As a result, we can utilize a large set of negative examples during training.

5 Experiments

Experiments are conducted using two object detection benchmark datasets. The first is the PASCAL VOC 2007 dataset [5], which contains 20 classes and 9963 images. We use the original train-test split ($\approx$ %50 each). The second is the SUN dataset [3], which contains 4367 training and 4317 test images and significantly more (107) classes. Most images in the VOC 2007 dataset contain at most five object classes, whereas many images in the SUN dataset contain around ten object classes.

In all experiments, the regularization parameter $\lambda = 0.01$ is used and each SetBoost strong classifier is trained with 100 decision tree weak classifiers. We report results with both the [26] and [27] versions of the detector of [14] as the baseline object detector. In our evaluation, we are most interested in the context-based improvement obtained for a given set of baseline detections rather than the absolute detection accuracies. For this reason, we always compare different methods using the same input baseline detections. For the other methods in comparison, we use either previously published results or use publicly available implementations. To quantify performance, the VOC object localization criteria [5] are used. The effectiveness of a context model is evaluated according to the change in the Average Precision (AP) of the detectors before and after the application of that particular model.



(a) (b)

Fig. 3. (a) Average frequency of each feature dimension chosen in the decision tree nodes for all classes in the VOC 2007 dataset. (b) Examples where horse and cat detectors give false positives on dogs in the VOC 2007 dataset.

5.1 Experiments on VOC 2007

For the VOC 2007 dataset, we use decision trees with (at most) 150 leaf nodes and the maximum set cardinality is 50.

Object Context. We first evaluate the effect of each feature dimension in the object context model. Note that each one of the learned object context models contains a set of decision trees. Fig. 3a shows the average frequency of selected feature dimensions in the first 25 weak classifiers among all classes. The first 20 features represent the detected object class and detection score, as defined in Section 3. The next six dimensions encode relative spatial positions, and the final two dimensions encode relative scores of object detection pairs.

The two most frequently chosen feature dimensions are the relative detection scores. An important reason is that relative scores can be used for inter-class and intra-class suppression. For instance, we observe that the AP detection scores for cat, cow, dog and horse classes improve significantly when our context model is applied (see Table 3). Since these animals have a relatively similar body shape, object detectors for these classes tend to return false positive detections for instances of each other class. Fig. 3b shows two images where horse and cat detectors give false positives on dogs. Based on the dog detection, the system suppresses false horse and cat detections in these images.

In Fig. 3a, we also observe that spatial relationship features (relative y, ..., overlap) are also frequently used. Therefore, the context model is not just an object class co-occurrence model; instead, a structured spatial layout of object classes is learned. For example, in Fig. 2 we observe that spatial relationships between "person-horse" and "person-bicycle" classes are learned.

We also note in Fig. 3a that the detection score features for the person class are frequently used in the object context models. This is probably related to the fact that the person class is the most dominant class in the VOC 2007 dataset.

Table 2. Comparison of SetBoost algorithm with PMK [7]. The baseline detector used is [26]. The column BL shows baseline AP scores and subsequent columns show the change in AP scores after applying different classifiers using the proposed object context representation.

	BL	PMK	BoW	Our		BL	PMK	BoW	Our		BL	PMK	BoW	Our
aero	28.6	1.4	0.4	**5.2**	cat	16.5	0.0	-2.5	**4.7**	person	35.8	-0.9	-2.8	**3.5**
bike	55.1	-0.2	-2.4	**2.3**	chair	16.5	-7.2	-0.6	**2.4**	plant	9.0	2.7	3.8	**5.2**
bird	0.6	1.4	1.4	**2.4**	cow	16.8	0.2	1.0	**2.0**	sheep	17.4	-0.7	1.4	**1.9**
boat	14.5	0.4	-0.2	**0.9**	dtable	24.6	0.3	**3.0**	0.9	sofa	22.7	1.0	-2.7	**3.7**
bottle	26.5	-3.8	-0.6	**3.4**	dog	5.0	**7.8**	0.5	5.3	train	34.0	0.6	-3.0	**4.3**
bus	39.7	-0.2	-3.9	**0.2**	horse	45.2	0.0	0.7	**5.1**	tv	38.3	-13.9	-13.3	**1.5**
car	50.1	-1.9	-3.6	**3.2**	mbike	38.3	-1.8	-3.0	**4.5**	AVG	26.8	-0.7	-1.3	**3.1**

Among true positive detections on the training set, 76% of all intra-class pairs are person-person, and 54% of all inter-class pairs involve person objects.

SetBoost vs. PMK vs. Bag-of-Words. Next, we evaluate SetBoost as opposed to SVM with Vocabulary-Guided Pyramid Match Kernel (PMK) [7], which is an alternative set classifier, and a Bag-of-Words (BoW) approach. We use PMK with 8 levels and branching factor set to 11. For BoW, we use k-means to construct a vocabulary of size 1000 per class and use SVMs with intersection kernel. The results are given in Table 2. We observe that the PMK and BoW also provide improvements in some of the classes, which assures that the proposed set-based context representation is effective. On the other hand, the performance of SetBoost classifier is clearly better for nearly all classes.

Scene Context. In learning the scene context model, decision trees with 5 leaf nodes are built with 50 weak classifiers. The scene-context has a vector representation, which reduces the required model complexity significantly. To avoid overfitting, we project GIST descriptors to 100 dimensions via PCA.

In the VOC 2007 dataset, there are significant scene changes and there is no dominantly occurring scene. As a result, evaluating scene context on its own yields poor performance, as one would expect: the change in the average AP score is -3.40 over the baseline. On the other hand, we see that scene context is complementary to object context and best performance is attained when the scene context and object context are used together. The average AP score increases by 3.65 when the combined model is used (see Table 3), compared to the 3.14 increase with object context model alone (see Table 2).

Comparison with Other Context Models. We first compare our approach against two state-of-the-art object context models [2] and [3]. For this purpose, we run all approaches on the outputs of the baseline detector [26].

Table 3 shows the baseline AP scores and the difference in AP scores ($\mathrm{AP}_{\mathrm{Diff}} = \mathrm{AP}_{\mathrm{After}} - \mathrm{AP}_{\mathrm{Before}}$) after applying the context models. The best improvement for each row of the table is shown in bold. As can be seen in Table 3, our object context model outperforms the state-of-the-art methods in 14 out of 20 classes, and attains best performance on average.

Table 3. Comparison of context models on VOC 2007 dataset using the baseline detections of [26]. Our method is using SetBoost with combined object and scene context model.

	BL	[2]	[3]	Our		BL	[2]	[3]	Our		BL	[2]	[3]	Our
aero	28.6	1.7	2.4	**5.4**	cat	16.5	.5	3.9	**5.0**	person	35.8	-2.8	0	**3.5**
bike	55.1	0	-4.2	**2.7**	chair	16.5	-2.8	**1.6**	0.0	plant	9.0	-1.1	**4.7**	4.2
bird	0.6	.1	2.3	**9.2**	cow	16.8	**1.2**	0.9	0.7	sheep	17.4	-2.3	**3.8**	3.1
boat	14.5	**1.4**	0.8	0.8	dtable	24.6	-.7	**2.3**	1.4	sofa	22.7	-0.7	2.8	**4.9**
bottle	26.5	0	-1.1	**3.2**	dog	5.0	.2	6.9	**7.9**	train	34.0	0.5	4.7	**4.9**
bus	39.7	-3.5	-0.2	**1.9**	horse	45.2	0.5	5.6	**5.9**	tv	38.3	0	-0.1	**0.3**
car	50.1	1.3	-0.4	**3.4**	mbike	38.3	1.1	2.2	**4.6**	AVG	26.8	-0.3	1.9	**3.6**

Table 4. Comparison of context model "20CMO+20OOI" of [6] with our approach on VOC 2007 using the baseline detections of [27]

	BL	[6]	CMO+Our		BL	[6]	CMO+Our		BL	[6]	CMO+Our
aero	28.9	2.6	**2.9**	cat	19.3	5.5	**5.9**	person	41.9	1.7	**1.8**
bike	59.5	**2.3**	0.4	chair	22.4	1.3	**2.7**	plant	12.2	**2.0**	1.1
bird	10.0	**2.4**	2.3	cow	25.2	**2.0**	1.7	sheep	17.9	1.7	**2.2**
boat	15.2	2.9	**4.7**	dtable	23.3	**7.4**	3.6	sofa	33.6	**4.9**	3.5
bottle	25.5	2.2	**2.6**	dog	11.1	2.6	**4.5**	train	45.1	**4.0**	3.9
bus	49.6	1.9	**2.0**	horse	56.8	**3.7**	3.1	tv	41.6	**2.7**	1.5
car	57.9	1.9	**2.1**	mbike	48.8	**2.3**	2.0	AVG	32.3	**2.9**	2.7

We also compare against [6], which combines a context model for unlabelled regions (CMO) with the object context model (OOI) of [14]. We introduce CMO scores into our model by simply concatenating per-class maximum CMO score to the scene context representation. Both models are applied to the output of the baseline detector [27]. As shown in Table 4, performance of the two approaches are comparable. Our method performs better in 10 out of 20 classes and [6] performs 0.2 points better on average.

Example outputs of our system are shown in Fig. 4. Images that include detections before and after contextual set-based rescoring shows how the irrelevant objects are eliminated and detection performance is improved.

5.2 Experiments with SUN Dataset

Since the number of objects and the amount of noise increases in SUN dataset, we increase the model complexity by doubling the decision tree size parameter (i.e., 300) and the maximum set cardinality (i.e., 100).

We compare with [3], which is the state-of-the-art method for this dataset. Table 5 shows the AP and AP_{Diff} scores averaged among 107 classes in the SUN dataset for the baseline detectors, [3] and our context model. When we apply our model, we get $AP_{Diff} = 1.68$ point improvement, whereas [3] results in $AP_{Diff} = 1.30$.

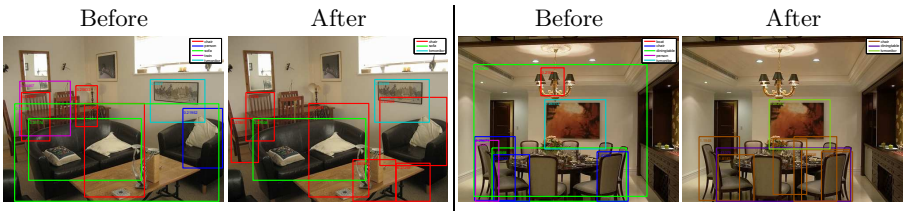


Fig. 4. Images before/after contextual rescoring. The eight top-scoring detections before and after applying the object context model are on the left and on the right respectively. In a livingroom image (left) the context model decreases the scores of false detections, such as a train and an overly large sofa, and it increases the scores of detected objects that are more likely to be correct. In a diningroom image (right) the model decreases the scores of a false person detection and a false boat detection, while it increases the scores of chair detections around the table.

Table 5. Comparison of our contextual object detection model versus the state-of-the-art on the SUN dataset

	Baseline	[3]	Our
Average AP	7.06	8.37	**8.75**
AP_{Diff} in Average AP	–	1.30	**1.68**

In the SUN dataset, the improvement obtained by applying either our context model or [3] is not as high as the improvement for the VOC 2007 dataset. We observe that baseline object detectors work poorly on the SUN dataset; 78% of them (83 out of 107) have an AP score less than 10 and 92% of them have an AP score less than 20. In contrast, on the VOC 2007 dataset, only 40% of the detectors have an AP score less than 20. Thus, although the SUN dataset has richer context in terms of contextual relations, extracting such contextual information is difficult because most of the baseline object detectors perform poorly. One might conclude that context models require sufficiently accurate baseline object detectors in order to gain significant improvement in context-base rescoring.

6 Conclusion

We have shown how object-object and object-scene context can be formulated in terms of sets. Our set-based model can learn the contextual relationships of objects without predefining the relationship types. The proposed SetBoost algorithm enables efficient learning of set classifiers. One advantage of SetBoost is its ability to use existing item classifiers (e.g., vector classifiers) to build set classifiers, without requiring intermediate representations. We also formulated a method for tuning decision trees for set-based classification within SetBoost. The effectiveness of our approach to contextual object recognition has been demonstrated on benchmark datasets.

In future work, we plan to address the handling of sets that contain items of varying dimensionalities. Another direction for future work is to handle multi-class classifiers directly.

Acknowledgments. This work was supported in part through US NSF grants 0855065, 0910908, and 1029430.

References

1. Blaschko, M.B., Lampert, C.H.: Object localization with global and local context kernels. In: BMVC (2009)
2. Desai, C., Ramanan, D., Fowlkes, C.: Discriminative models for multi-class object layout. In: ICCV (2009)
3. Choi, M.J., Lim, J.J., Torralba, A., Willsky, A.S.: Exploiting hierarchical context on a large database of object categories. In: CVPR (2010)
4. Gemert, J.C.V., Snoek, C.G., Veenman, C.J., Smeulders, A.W., Geusebroek, J.M.: Comparing compact codebooks for visual categorization. CVIU (2010)
5. Everingham, M., Van Gool, L., Williams, C.K.I., Winn, J., Zisserman, A.: The PASCAL Visual Object Classes Challenge (2007)
6. Li, C., Parikh, D., Chen, T.: Extracting adaptive contextual cues from unlabeled regions. In: ICCV (2011)
7. Grauman, K., Darrell, T.: Approximate correspondences in high dimensions. In: NIPS (2007)
8. Heitz, G., Koller, D.: Learning Spatial Context: Using Stuff to Find Things. In: Forsyth, D., Torr, P., Zisserman, A. (eds.) ECCV 2008, Part I. LNCS, vol. 5302, pp. 30–43. Springer, Heidelberg (2008)
9. Liu, C., Yuen, J., Torralba, A.: Nonparametric scene parsing: Label transfer via dense scene alignment. In: CVPR (2009)
10. Galleguillos, C., Rabinovich, A., Belongie, S.: Object categorization using co-occurrence, location and appearance. In: CVPR (2008)
11. Oliva, A., Torralba, A.: Modeling the shape of the scene: A holistic representation of the spatial envelope. IJCV 42 (2001)
12. Hoiem, D., Efros, A., Hebert, M.: Putting objects in perspective. IJCV (2008)
13. Galleguillos, C., Belongie, S.: Context based object categorization: A critical survey. CVIU 114 (2010)
14. Felzenszwalb, P., McAllester, D., Ramanan, D., Grishick: Object detection with discriminatively trained part based models. PAMI 32 (2010)
15. Kondor, R., Jebara, T.: A kernel between sets of vectors. In: ICML (2003)
16. Cuturi, M., Vert, J.: Semigroup kernels on finite sets. In: NIPS (2005)
17. Lyu, S.: Mercer kernels for object recognition with local features. In: CVPR (2005)
18. Csurka, G., Dance, C., Fan, L., Willamowski, J., Bray, C.: Visual categorization with bags of keypoints. In: Workshop on Stat. Learning in Comp. Vision (2004)
19. Moosmann, F., Nowak, E., Jurie, F.: Randomized clustering forests for image classification. PAMI 30 (2008)
20. Yang, L., Jin, R., Sukthankar, R., Jurie, F.: Unifying discriminative visual codebook generation with classifier training for object recognition. In: CVPR (2008)
21. Dollár, P., Babenko, B., Belongie, S., Perona, P., Tu, Z.: Multiple Component Learning for Object Detection. In: Forsyth, D., Torr, P., Zisserman, A. (eds.) ECCV 2008, Part II. LNCS, vol. 5303, pp. 211–224. Springer, Heidelberg (2008)

22. Mason, L., Baxter, J., Bartlett, P., Frean, M.: Boosting algorithms as gradient descent in function space. In: NIPS (1999)
23. Byrd, R.H., Lu, P., Nocedal, J., Zhu, C.: A limited memory algorithm for bound constrained optimization. SIAM J. on Scientific Comp. 16 (1995)
24. Freund, Y., Schapire, R.E.: A Decision-Theoretic generalization of On-Line learning and an application to boosting. J. of Comp. and Sys. Sci. 55 (1997)
25. Friedman, J.H.: Stochastic gradient boosting. Comp. Stat. and Data Analysis 38 (2002)
26. http://people.cs.uchicago.edu/~pff/latent-release3.1/
27. http://people.cs.uchicago.edu/~pff/latent-release4/