Exploiting Hierarchical Context on a Large Database of Object Categories

Myung Jin Choi, Joseph J. Lim, Antonio Torralba, Alan S. Willsky
Massachusetts Institute of Technology
myungjin@mit.edu, lim@csail.mit.edu, torralba@csail.mit.edu, willsky@mit.edu

Abstract

There has been a growing interest in exploiting contextual information in addition to local features to detect and localize multiple object categories in an image. Context models can efficiently rule out some unlikely combinations or locations of objects and guide detectors to produce a semantically coherent interpretation of a scene. However, the performance benefit from using context models has been limited because most of these methods were tested on datasets with only a few object categories, in which most images contain only one or two object categories. In this paper, we introduce a new dataset with images that contain many instances of different object categories and propose an efficient model that captures the contextual information among more than a hundred of object categories. We show that our context model can be applied to scene understanding tasks that local detectors alone cannot solve.

1. Introduction

Standard single-object detectors [3, 5] focus on locally identifying a particular object category. In order to detect multiple object categories in an image, we need to run a separate detector for each object category at every spatial location and scale. Since each detector works independently from others, the outcome of these detectors may be semantically incorrect.

Even if we have perfect local detectors that correctly identify all object instances in an image, some tasks in scene understanding require an explicit context model, and cannot be solved with local detectors alone. An example of this is detecting unexpected objects that are out of their normal context. Figure 1 shows one example of images in which an object is out of context. These scenes attract a human's attention since they don't occur often in daily settings. Understanding how objects relate to each other is important to answer queries such as *find some funny pictures* or *where can I leave the keys so that I can find them later?*

A simple form of contextual information is a co-occurrence frequency of a pair of objects. Rabinovich et al. [19] use local detectors to first assign an object label to each image segment, and adjusts these labels using a conditional random field. [7] and [8] extend this approach to encode spatial relationships between a pair of objects. In [7], spatial relationships are quantized to four prototypical relationships - above, below, inside and around, whereas in [8] a non-parametric map of spatial priors are learned for each pair of objects. Torralba et al. [24] combine boosting and CRF's to first detect easy objects (e.g., a monitor) and pass the contextual information to detect other more difficult objects (e.g., a keyboard). [25] uses both image patches and their probability maps estimated from classifiers to learn a contextual model, and iteratively refines the classification results by propagating the contextual information. [4] combines individual classifiers by using spatial interactions between object detections in a discriminative manner.

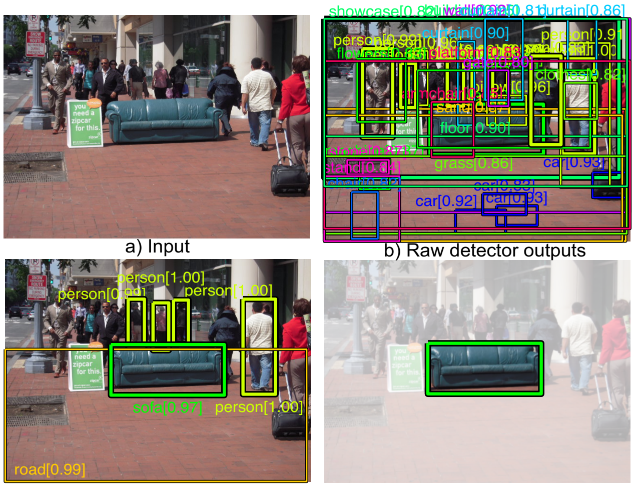

Figure 1. Detecting objects in and out of context. a) Input image, b) Output of 107 class detectors. With so many classes many false alarms appear on the image providing a useless scene interpretation. c) Output of our context model. d) Most unexpected object in the image. This output can not be produced by object detectors alone, even if they are perfect. Detecting out of context objects requires modeling what the expected scene configurations are.

Contextual information may be obtained from coarser, global features as well. Torralba [23] demonstrates that a global image feature called a "gist" can predict the presence or absence of objects and their locations without running an object detector. [16] extend this approach to combine patch-based local features and the gist feature. Heitz and Koller [11] combine a sliding window method and unsupervised image region clustering to leverage "stuff" such as the sea, the sky, or a road to improve object detection. [10] introduces a cascaded classification model, which links scene categorization, multi-class image segmentation, object detection, and 3D reconstruction.

Hierarchical models can incorporate both local and global images features. [9] uses multiscale conditional random fields to combine local classifiers with regional and global features. Sudderth et al. [22] model the hierarchy of scenes, objects and parts using hierarchical Dirichlet processes, which encourage scenes to share objects, objects to share parts, and parts to share features. Parikh and Chen [17] learn a hierarchy of objects in an unsupervised manner, under the assumption that each object appears exactly once in all images. Hierarchical models are also common within grammar models for scenes [18, 14] and they have been shown to be very flexible to represent complex relationships. Bayesian hierarchical models also provide a powerful mechanism to build generative scene models [15].

In this work, we model object co-occurrences and spatial relationships using a tree graphical model. We combine this prior model of object relationships with local detector outputs and global image features to detect and localize all instances of multiple object categories in an image. Enforcing tree-structured dependencies among objects allows us to learn our model for more than a hundred of object categories and apply it to images efficiently. Even though we do not explicitly impose a hierarchical structure in our learning procedure, the tree organizes objects in a natural hierarchy.

In order to exploit contextual information, it is important to have many different object categories present simultaneously in an image, with a large range of difficulties (from large to small objects). Here we introduce a new dataset (SUN09), with more than 200 object categories in a wide range of scene categories, which is suitable for contextual information.

2. A new dataset for context based recognition

We introduce a new dataset (SUN09) suitable for leveraging the contextual information. The dataset contains 12.000 annotated images covering a large number of scene categories (indoor and outdoors) with more than 200 object categories and 152.000 annotated object instances. SUN09 has been annotated using LabelMe [21] by a single annotator and verified for consistency.

Figure 2 shows statistics of out dataset and compares them with PASCAL07. The Pascal dataset provides an excellent framework for evaluating object detection algorithms. However, this dataset, as shown in Figure 2, is not suitable to test context-based object recognition algorithms. The PASCAL dataset contains 20 object classes, but more than 50% of the images contain only a single object class. MSRC [26] procides more co-ocurring objects but it only contains 23 object classes. Contextual information is most useful when many object categories are present simultaneously in an image, with some object instances that are easy to detect (i.e. large objects) and some instances that are hard to detect (i.e. small objects). The average PASCAL bounding box occupies 20% of the image. On the other hand, in our dataset, the average object size is 5% of the image size, and a typical image contains 7 different object categories. Figure 2 (c-d) show typical images from each dataset.

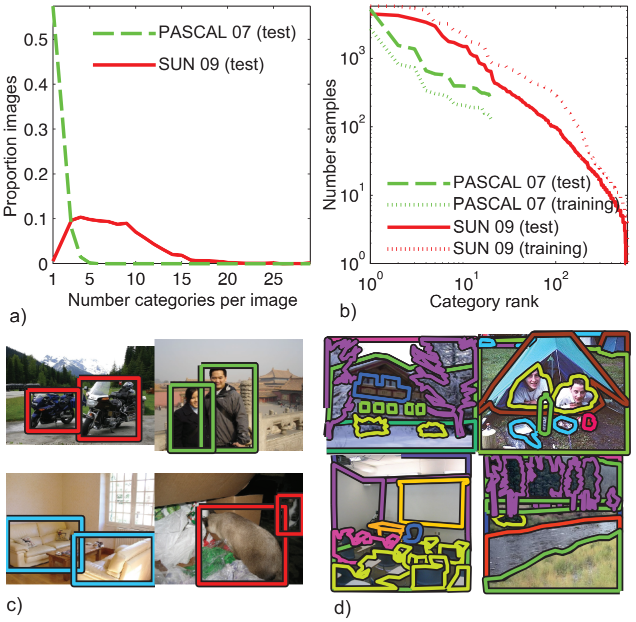

Figure 2. Comparison PASCAL 07 and our dataset (SUN09). a) Histogram of number of object categories present in each image. b) Distribution of training and test samples per each object category. c) 4 examples from the set of typical PASCAL images. A typical pascal image contains two instances of a single object category, and objects occupy 20% of the image. d) 4 examples from the set of typical SUN images. A typical SUN image has 7 object categories (with around 14 total annotated objects) and occupy a wide range of sizes (average 5%).

3. Tree-structured contextual model

We use a tree graphical model to learn dependencies among object categories. [19] uses a fully-connected CRF to model object dependencies, which is computationally expensive for modeling relationships among many object categories. [16] models dependencies among objects using scene-object relationships, and assumes that objects are independent conditioned on the scene type, which may ignore direct dependencies among objects. Our tree provides a richer representation of object dependencies and enables efficient inference and learning algorithms. In this section, we describe a prior model that captures co-occurrence statistics and spatial relationships among objects, and explain how global image features and local detector outputs can be integrated into the framework as measurements.

3.1. Prior model

3.1.1 Co-occurrences prior

A simple yet effective contextual information is the co-occurence of object pairs. We encode the co-occurrence statistics using a binary tree model. Each node b_i in a tree represents whether the corresponding object i is present or not in an image. The joint probability of all binary variables are factored according to the tree structure:

$$p(b) = p(b_{root}) \prod_i p(b_i | b_{pa(i)}) \tag{1}$$

where $pa(i)$ is the parent of node i. Note that the parent-child pairs may have either positive (e.g., `floor` and `wall` co-occur often) or negative (e.g., `floor` never appears with `sky`) relationships.

3.1.2 Spatial prior

Spatial location representation Objects often appear at specific relative positions to one another. For example, a computer screen, a keyboard, and a mouse generally appear in a fixed arrangement. We capture such spatial relationships by adding location variables to the tree model. Let ℓ_x, ℓ_y be the x,y coordinate of the center of the bounding box, and ℓ_w, ℓ_h be the width and height of the box. We assume that the image height is normalized to one, and that $\ell_x = 0, \ell_y = 0$ is the center of the image. The expected distance between centers of objects depends on the size of the objects - if a keyboard and a mouse are small, the distance between the centers should be small as well. Constellation model [6] achieves scale invariance by transforming the position information to a scale invariant space. Hoiem et al. [13] relate scale changes to an explicit 3D information. We take Hoeim et.al 's approach and apply the following coordinate transformations to represent object locations in the 3D-world coordinate:

$$L_x = \frac{\ell_x}{\ell_h} H_i, \;\; L_y = \frac{\ell_y}{\ell_h} H_i, \;\; L_z = \frac{f}{\ell_h} H_i \tag{2}$$

where f is the distance from observer to the image plane, which we set to 1, and L_z is the distance between the observer and the object. H_i is the physical height of an object i, which is assumed to be constant. These constants could be inferred from the annotated data using the algorithm in [12]. Instead, we model the object sizes by manually encoding real object sizes (e.g., person = 1.7m, car = 1.5m). We assume that all objects have fixed aspect ratios.

Prior on spatial locations The x-coordinates of objects varies considerably from one image to another, and is uninformative in general [23]. Thus, we ignore L_x and only consider L_y and L_z to capture vertical location and scale relationships. We assume that L_y's and L_z's are independent, i.e., the vertical location of an object is independent from its distances from the image plane. While we model L_y as jointly Gaussian, we model L_z as a log-normal distribution since it is always positive and is more heavily distributed around small values. We can redefine a location variable for object category i as $L_i = (L_y, \log L_z)$ and model L_i's as jointly Gaussian. If there are multiple instances of object category i in an image, L_i represents the median location of all instances.

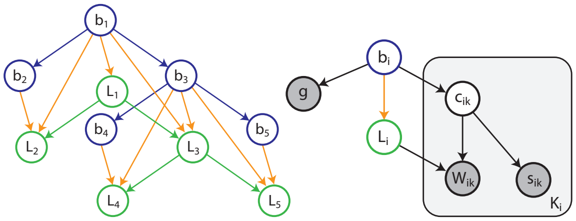

Figure 3. (Left) Prior model relating object presence variables b_i's and location variables L_i. (Right) Measurement model for object i. The gist descriptor g represents global image features, and local detector provides candidate window locations W_{ik} and scores s_{ik}. c_{ik} indicates whether the window is a correct detection or not.

We assume that when conditioned on the presence variable b, the dependency structure of the L_i's has the same tree structure as our binary tree:

$$p(L|b) = p(L_{root}|b_{root}) \prod_i p(L_i | L_{pa(i)}, b_i, b_{pa(i)}), \tag{3}$$

where each edge potential $p(L_i | L_{pa(i)}, b_i, b_{pa(i)})$ encodes the distribution of a child location conditioned on its parent location and the presence/absence of both child and parent objects. We use three different Gaussian distributions to define $p(L_i | L_{pa(i)}, b_i, b_{pa(i)})$ for each parent-child pair. When both child and parent objects are present ($b_i = 1, b_{pa(i)} = 1$), the expected location of the child object i is determined by the location of its parent $L_{pa(i)}$. When the object is present but its parent object is not ($b_i = 1, b_{pa(i)} = 0$), then L_i is independent from $L_{pa(i)}$. When an object is not present ($b_i = 0$), we assume that its location is independent from all other object locations and let L_i represent the average location of the object i across all images.

Figure 3 shows the graphical model of the presence variable b and the location variable L. Combining (1) and (3), the joint distribution of all binary and Gaussian variables can be represented as follows:

$$\begin{aligned} p(b, L) =& p(b)p(L|b) = p(b_{root})p(L_{root}) \\ & \times \prod_i p(b_i | b_{pa(i)}) p(L_i | L_{pa(i)}, b_i, b_{pa(i)}). \end{aligned} \tag{4}$$

If we combine b_i and L_i as a single variable O_i, we observe that $p(O)$ also has a tree structure. Even though the full graphical model with respect to b and L is not a tree, the dependency between objects forms a tree structure. In the rest of the paper, we refer to this model as a prior tree model, assuming that each node in the tree corresponds to O_i.

3.2. Measurement model

3.2.1 Incorporating global image features

In addition to incorporating relationships among objects, we introduce gist [23] as a measurement for each presence variable b_i, to incorporate global image features into our model. Since the gist is a high-dimensional vector, we use logistic regression to fit $p(b_i|g)$ [16], from which we estimate the likelihoods $p(g|b_i)$ indirectly using $p(g|b_i) = p(b_i|g)p(g)/p(b_i)$ to avoid overfitting.

3.2.2 Integrating local detector outputs

In order to detect and localize object instances in an image, we first apply off-the-shelf single-object detectors and obtain a set of candidate windows for each object category. Let i denote an object category and let k index candidate windows generated by baseline detectors. Each detector output provides a score s_{ik} and a bounding box, to which we apply the coordinate transformation in (2) to get the location variable $W_{ik} = (L_y, \log L_z)$. We assign a binary variable c_{ik} to each window to represent whether it is a correct detection ($c_{ik} = 1$) or a false positive ($c_{ik} = 0$). Figure 3 shows the measurement model for object i to integrate gist and outputs from local detectors into our prior model, where we used plate notations to represent K_i different candidate windows.

We sort the baseline score for each object category and assign candidate window index k so that s_{ik} is the k-th highest score for category i. The probability of correct detection $p(c_{ik} = 1|b_i = 1)$ is trained from the training set. If object i is not present, then all the candidate windows are false positives: $p(c_{ik} = 1|b_i = 0) = 0$.

The distribution of scores depends on whether the window is a correct detection or a false positive. We could fit a truncated Gaussian distribution for $p(s_{ik}|c_{ik} = 0)$ and for $p(s_{ik}|c_{ik} = 1)$. Estimating parameters can be unreliable if there are only few samples with the correct detection. To address this issue, we use logistic regression to train $p(c_{ik}|s_{ik})$ and compute the likelihood using $p(s_{ik}|c_{ik}) = p(c_{ik}|s_{ik})p(s_{ik})/p(c_{ik})$.

If a candidate window is a correct detection of object i ($c_{ik} = 1$), then its location W_{ik} is a Gaussian vector with mean L_i, the expected location of object i:

$$p(W_{ik}|c_{ik} = 1, L_i) = \mathcal{N}(W_{ik}; L_i, \Lambda_i) \qquad (5)$$

where Λ_i is the covariance around the predicted location [16]. If the window is a false positive ($c_{ik} = 0$), W_{ik} is independent from L_i and has a uniform distribution.

4. Alternating inference on trees

Given the gist g, candidate window locations $W \equiv \{W_{ik}\}$ and their scores $s \equiv \{s_{ik}\}$, we infer the presence of objects $b \equiv \{b_i\}$, the correct detections $c \equiv \{c_{ik}\}$, and expected locations of all objects $L \equiv \{L_i\}$, by solving the following optimization problem:

$$\hat{b}, \hat{c}, \hat{L} = \underset{b,c,L}{\operatorname{argmax}}\, p(b, c, L|g, W, s) \qquad (6)$$

Although our overall model is a tree if we consider b_i and L_i as a single node, the exact inference is complicated since there are both binary and Gaussian variables in the model. For efficient inference, we leverage the tree structures embedded in the prior model. Specifically, conditioned on b and c, the location variables L forms a Gaussian tree. On the other hand, conditioned on L, the presence variables b and the correct detection variables c together form a binary tree. For each of these trees, there exists efficient inference algorithms [1]. Therefore, we infer b, c and L in an alternating manner.

In our first iteration, we ignore the location information W, and sample[1] b and c conditioned only on the gist g and the candidate windows scores s: $\hat{b}, \hat{c} \sim p(b, c|s, g)$. Conditioned on these samples, we infer the expected locations of objects $\hat{L} = \operatorname{argmax}_L p(L|\hat{b}, \hat{c}, W)$ using belief propagation on the resulting Gaussian tree. Then conditioned on the estimate of locations $\hat{L}$, we re-sample b and c conditioned also on the window locations: $\hat{b}, \hat{c} = \operatorname{argmax}_{b,c} p(b, c|s, g, \hat{L}, W)$, which is equivalent to sampling from a binary tree with node and edge potentials modified by the likelihoods $p(\hat{L}, W|b, c)$. In this step, we encourage pairs of objects or windows in likely spatial arrangements to be present in the image.

We iterate between sampling on the binary tree and inference on the Gaussian tree, and select the samples $\hat{b}$ and $\hat{c}$ with the highest likelihood. We use 4 different starting samples each with 3 iterations in our experiments. Our inference procedure is efficient even for models with hundreds of objects categories and thousands of candidate windows. For the SUN dataset, it takes about 0.5 second in MATLAB to produce estimates from one image.

5. Learning

We learn the dependency structure among objects from a set of fully labeled images. The Chow-Liu algorithm [2] is a simple and efficient way to learn a tree structure that maximizes the likelihood of the data: the algorithm first computes empirical mutual information of all pairs of variables using their sample values. Then, it finds the maxi-

[1] We can also compute the MAP estimates of these binary variables efficiently, but starting from the MAP estimates and iterating between the binary and Gaussian trees typically leads to a local maximum that is close from the initial MAP estimates.

mum weight spanning tree with edge weights equal to the mutual information. We learn the Chow-Liu tree using the statistics of b_i's computed from a set of labeled images. We pick a root node arbitrarily once a tree structure is learned. Even with more than 100 objects and thousands of training images, a tree model can be learned in a few seconds in MATLAB.

Figure 7 shows a tree structure learned from the SUN09 dataset. We selected `sky` to be the root of the tree. It is interesting to note that even though the Chow-Liu algorithm is simply selecting strong pairwise dependencies, our tree organizes objects in a natural hierarchy. For example, a subtree rooted at `building` has many objects that appear in street scenes, and the subtree rooted at `sink` contains objects that would commonly appear in a kitchen. Thus, many non-leaf nodes act as if they are representing coarser scale meta-objects or scene categories. In other words, the learned tree captures the inherent hierarchy among objects and scenes, resulting in significant improvements in object recognition and scene understanding as demonstrated in 6.

6. Results

6.1. Recognition performance on PASCAL07

Context learned from training images We train the context model on 4367 images from the training set. Figure 4.a shows the tree learned for this dataset. The model correctly captures important contextual relationships among objects (co-ocurrences and relative spatial locations). Figure 4.b shows a few samples from the joint model of 3D locations, illustrating the relative spatial relationship among objects. Our model correctly learns that most training images contain 1 or few objects, and that the spatial information embedded in PASCAL07 data is limited.

Object recognition performance Table 1 provides the average precision-recall (APR) for the object localization task, obtained for the objects in the pascal dataset using different models and compares the results with one of the state of the art models at this task that also incorporates contextual information [4]. For the baseline detector, we use the detector in [5], which is based on the mixture of multiscale deformable part model. There is a slight advantage in incorporating context, but not a huge improvement. As discussed in Section 2, this dataset contains very little contextual information among objects and the performance benefit from incorporating the contextual information is small. We show in the next section that the contextual information does improve the performance significantly when we use the new dataset SUN09. One thing to note is that the best achievable performance is limited by the recall of the detector since context models are only used to enhance the scores of the bounding boxes (or segments) proposed by a detector.

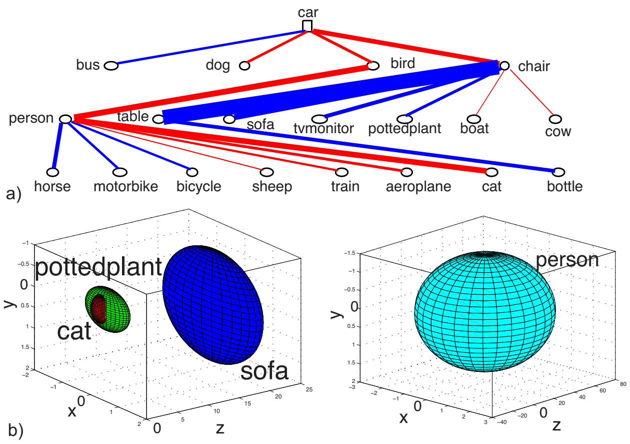

Figure 4. a) Model learned from PASCAL 07. Red edges correspond to negative correlations between classes. The thickness of each edge represents the strength of the link. b) 3D samples generated from the context model.

Figure 5a. compares performance of our context model to that of the baseline detector for the localization task (i.e., detecting the correct bounding box). We look at N most confident detections in each image and check whether they are all correct. For the baseline detector, we use a logistic regression to compute the probability of correct detection based on the detector score. For the context model, we compute the probability of correct detection given gist and detector outputs (i.e. $p(c_{ik} = 1|s, g, W)$) using the efficient inference algorithm described in Section 4. The numbers on top of the bars indicate the number of images that contain at least N ground-truth object instances.

Figure 5b. compares the baseline and the context model for the presence predication task (i.e., is the object present in the scene?). We compute the probability of each object category being present in the image, and check whether the top N object categories are all correct. The most confident detection for each object category is used for the baseline detector. For the context model, we compute the probability of each object class being present in the image (i.e. $p(b_i = 1|s, g, W)$). The numbers on top of the bars indicate the number of images that contain at least N different ground-truth object categories. Note that the number of images drops significantly as N gets larger since most images in PASCAL contain only one or two object categories. The context model outperforms the baseline significantly for the presence prediction task.

6.2. Recognition performance on SUN09 dataset

We divide the SUN09 dataset into two sets of equal sizes, one for training and the other for testing. Each set has the same number of images per scene category. In order to have enough training samples for the baseline detectors [5], we annotated an additional set of 26.000 objects using Amazon Mechanical Turk. This set consists of images with a single annotated object. This set was used only for training the

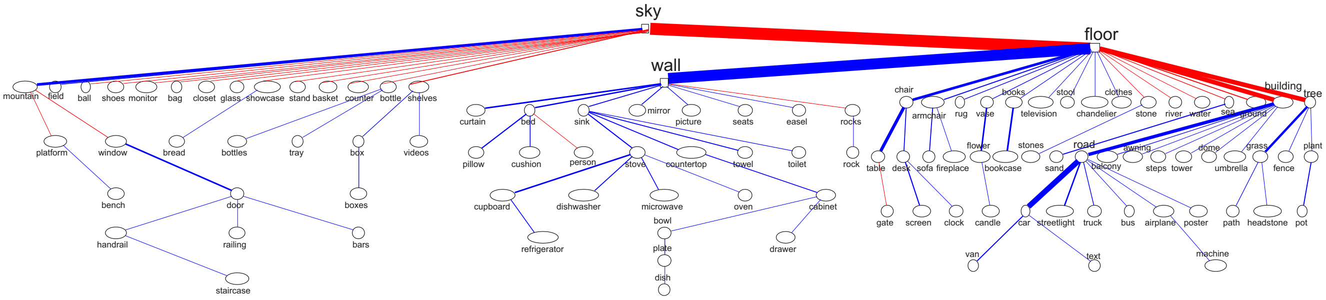

Figure 7. Model learned for SUN 09. Red edges denote negative correlation between classes. The thickness of each edge represents the strength of the link.

Category	Baseline	Gist	Context	Baseline in [4]	[4]	Bound
aeroplane	28.12	31.30	**32.05**	27.80	28.80	50.88
bicycle	51.52	50.79	50.56	55.90	**56.20**	58.76
bird	1.93	0.75	0.89	1.40	**3.20**	27.45
boat	13.85	**15.06**	14.90	14.60	14.20	28.14
bottle	23.44	25.58	25.28	25.70	**29.40**	40.51
bus	**38.87**	35.83	36.98	38.10	38.70	47.89
car	47.01	46.74	46.74	47.00	**48.70**	65.95
cat	14.73	16.72	**18.93**	15.10	12.40	48.60
chair	16.01	17.91	**18.12**	16.30	16.00	49.08
cow	**18.24**	18.07	18.22	16.70	17.70	36.89
diningtable	21.01	23.18	22.93	22.80	**24.00**	30.58
dog	10.73	11.26	**12.43**	11.10	11.70	46.22
horse	43.22	45.32	**47.29**	43.80	45.00	69.54
motorbike	40.27	40.99	**41.87**	37.30	39.40	59.69
person	35.46	34.77	35.46	35.20	**35.50**	58.92
pottedplant	14.90	**16.55**	15.67	14.00	15.20	43.75
sheep	19.37	21.77	**21.81**	16.90	16.10	35.13
sofa	**20.56**	19.43	20.40	19.30	20.10	42.67
train	37.74	37.43	**38.80**	31.90	34.20	61.35
tvmonitor	37.00	34.27	35.75	**37.30**	35.40	54.87
AVERAGE	26.70	27.19	**27.75**	26.41	27.10	47.84

Table 1. Average precision-recall. Baseline) baseline detector [5]; Gist) baseline and gist [20]; Context) our context model; [4]) results from [4] (the baseline in [4] is the same as our baseline, but performances slightly differ); Bound) Maximal APR that can be achieved given current max recall.

baseline detector and not for learning the tree model.

In this experiment we use 107 object detectors. These detectors span from regions (e.g., road, sky, buildings) to well defined objects (e.g., car, sofa, refrigerator, sink, bowl, bed) and highly deformable objects (e.g., river, towel, curtain). The database contains 4317 test images. Objects have a large range of difficulties due to variations in shape, but also in sizes and frequencies. The distribution of objects in the test set follows a power law (the number of instances for object k is roughly $1/k$) as shown in Figure 2.

Context learned from training images Figure 7 shows the learned tree relating the 107 objects. A notable difference from the tree learned for PASCAL07 (Figure 4) is that the proportion of positive correlations is larger. In the tree learned from PASCAL07, 10 out of 19 edges, and 4 out of the top 10 strongest edges have negative relationships. In contrast, 25 out of 106 edges and 7 out of 53 ($\approx 13\%$) strongest edges in the SUN tree model have negative relationships. In PASCAL07, most objects are related by repulsion because most images contain only few categories. In SUN09, there is a lot more opportunities to learn positive correlations between objects. From the learned tree, we can see that some objects take the role of dividing the tree ac-

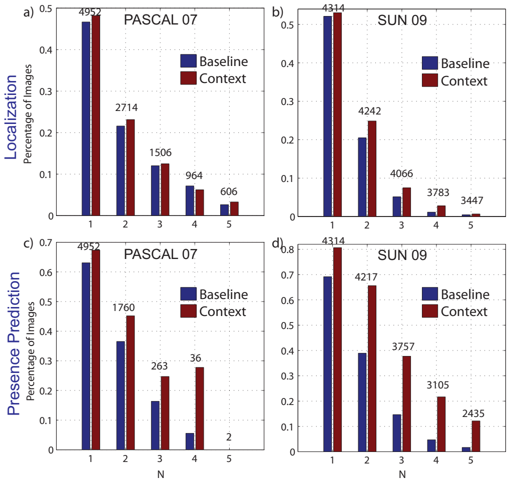

Figure 5. Summary of results for PASCAL 07 and SUN 09. a-b) Percentage of images in which the top N most confident detections are all correct. The numbers on top of the bars indicate the number of images that contain at least N ground-truth object instances. c-d) Percentage of images in which the top N most confident object presence predictions are all correct. The numbers on top of the bars indicate the number of images that contain at least N different ground-truth object categories.

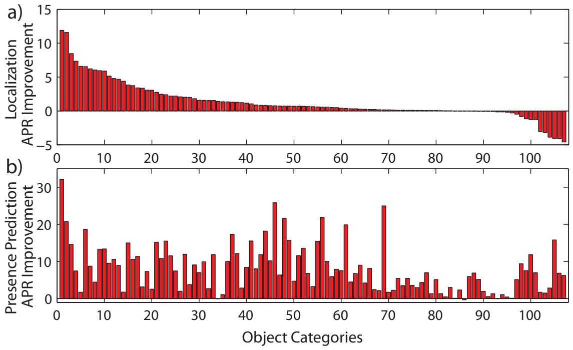

Figure 6. Improvement of context model over the baseline. Object categories are sorted by the improvement in the localization task.

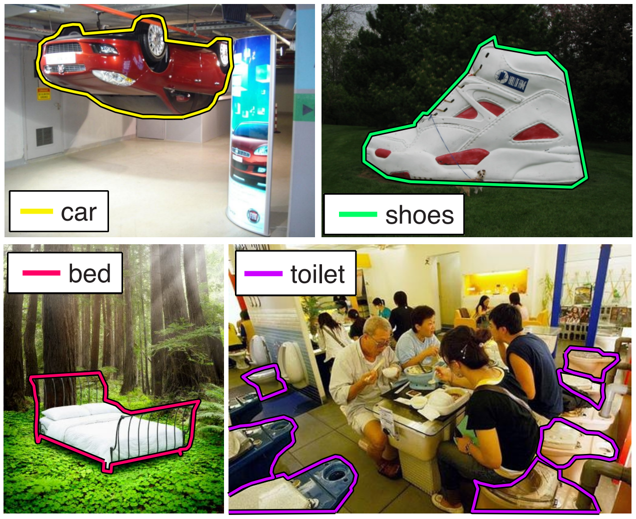

Figure 9. Four examples of objects out of context (wrong pose, wrong scale, wrong scene wrong co-occurrence). The segments show the objects selected by the contextual model (the input of the system are the true segmentations and labels, and the model task is to select which objects are out of context).

cording to the scene category as described in Section 5. For instance, `floor` separates indoor and outdoor objects.

Object recognition performance Despite the high variance in object appearances, the baseline detectors have a reasonable performance. Figure 5(b,d) show localization and presence prediction results on SUN09. Note that the context model improve the image annotation results significantly: as shown in Figure 5(d), among the 3757 images that contain at least three different object categories, the three most confident objects are all correct for 38% of images (and only 15% without context).

Figure 6 show the improvement in average precision-recall (APR) for each object category. Due to the large number of objects in our database, there are many objects that benefit in different degrees from context. Six objects with the largest improvement with context for the localization task are floor (+11.88), refrigerator (+11.58), bed (+8.46), seats(+7.34), monitor (+6.57), and road (+6.55). The overall localization APR averaged over all object categories is 7.06 for the baseline and 8.37 for the context model. Figure 8 shows some image annotation results. For each image, only the six most confident detections are shown.

6.3. Detecting images out of context

Figure 9 shows some images with one or more objects in an unusual setting such as scale, position, or scene. Objects that are out-of-context generally have different appearances or viewpoints from typical training examples, making local detectors perform poorly. Even if we have perfect local detectors, or ground-truth labels, we need contextual information to identify out-of-context scenes, which is not available from local detector outputs.

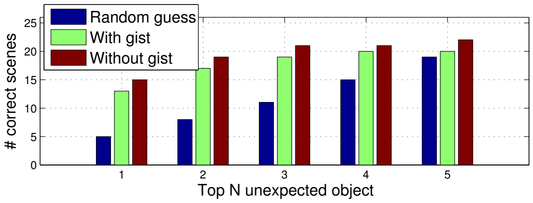

Figure 10. Performance on detecting objects out of context.

In this section, we present preliminary results in detecting objects out-of-context. For this task, we created a database of 26 images with one or more objects that are out of their normal context. In each test, we assume that we have ground-truth object labels for all objects in the scene, except for the one under the test. Among all objects present in the image, we picked an object label with the lowest probability conditioned on all other (ground-truth) object labels in the scene using our context model. Figure 9 shows some examples where the context model correctly identifies objects that are the most unexpected object in the scene.

Figure 10 shows the number of images that at least one out-of-context object was included in the top N unexpected objects estimated by the context model. It is interesting to note that using gist may hurt the performance of detecting images out of context. This is due to the fact that those objects may change global features of an image, biasing gist to favor that object.

7. Conclusion

We present a new dataset and an efficient methodology to model contextual information among over 100 object categories. The new dataset SUN09 contains richer contextual information compared to Pascal07, which was originally designed for training object detectors. We demonstrate that the contextual information learned from SUN09 significantly improves the accuracy of object recognition tasks, and can even be used to identify out-of-context scenes. The tree-based context model enables an efficient and coherent modeling of regularities among object categories, and can easily scale to capture dependencies of over 100 object categories. Our experiments provide compelling evidence that rich datasets and modeling frameworks that incorporate contextual information can be more effective at a variety of computer vision tasks such as object classification, object detection, and scene understanding.

Acknowledgment

This research was partially funded by Shell International Exploration and Production Inc., by NSF Career Award (ISI 0747120), and by the Air Force Office of Scientific Research under Award No.FA9550-06-1-0324. Any opinions, findings, and conclusions or recommendations expressed in

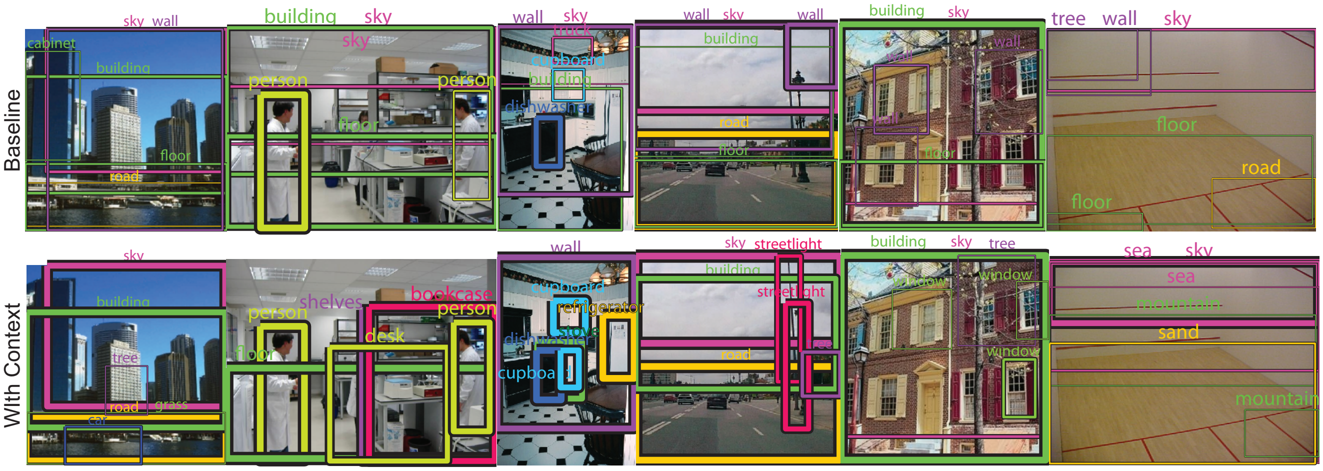

Figure 8. Examples of scenes showing the six most confident detections with and without context. The figure shows successful examples of using context as well as failures.

this publication are those of the author(s) and do not necessarily reflect the views of the Air Force.

References

[1] C. M. Bishop. *Pattern Recognition and Machine Learning*. Springer, 2006. 4

[2] C. K. Chow and C. N. Liu. Approximating discrete probability distributions with dependence trees. *IEEE TIT*, 1968. 4

[3] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *CVPR*, 2005. 1

[4] C. Desai, D. Ramanan, and C. Fowlkes. Discriminative models for multi-class object layout. In *ICCV*, 2009. 1, 5, 6

[5] P. Felzenszwalb, D. McAllester, and D. Ramanan. A discriminatively trained, multiscale, deformable part model. In *CVPR*, 2008. 1, 5, 6

[6] R. Fergus, P. Perona, and A. Zisserman. Object class recognition by unsupervised scale-invariant learning. In *CVPR*, 2003. 3

[7] C. Galleguillos, A. Rabinovich, and S. Belongie. Object categorization using co-occurrence, location and appearance. In *CVPR*, 2008. 1

[8] S. Gould, J. Rodgers, D. Cohen, G. Elidan, and D. Koller. Multi-class segmentation with relative location prior. *IJCV*, 80(3):300–316, 2007. 1

[9] X. He, R. S. Zemel, and M. A. C.-P. nán. Multiscale conditional random fields for image labeling. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'04)*, 2004. 2

[10] G. Heitz, S. Gould, A. Saxena, and D. Koller. Cascaded classification models: Combining models for holistic scene understanding. In *NIPS*, 2008. 2

[11] G. Heitz and D. Koller. Learning spatial context: Using stuff to find things. In *Proc. 10th European Conference on Computer Vision*, 2008. 2

[12] D. Hoiem, A. Efros, and M. Hebert. Automatic photo popup. In *SIGGRAPH*, 2005. 3

[13] D. Hoiem, A. Efros, and M. Hebert. Putting objects in perspective. In *CVPR*, 2006. 3

[14] Y. Jin and S. Geman. Context and hierarchy in a probabilistic image model. In *CVPR*, pages 2145–2152, Washington, DC, USA, 2006. IEEE Computer Society. 2

[15] L.-J. Li, R. Socher, and L. Fei-Fei. Towards total scene understanding:classification, annotation and segmentation in an automatic framework. In *CVPR*, 2009. 2

[16] K. P. Murphy, A. Torralba, and W. T. Freeman. Using the forest to see the trees: a graphical model relating features, objects and scenes. In *NIPS*, 2003. 2, 4

[17] D. Parikh and T. Chen. Hierarchical semantics of objects (hsos). In *IEEE International Conference in Computer Vision (ICCV)*, volume 2008, 2007. 2

[18] J. Porway, K. Wang, B. Yao, and S. C. Zhu. A hierarchical and contextual model for aerial image understanding. *CVPR*, pages 1–8, 2008. 2

[19] A. Rabinovich, A. Vedaldi, C. Galleguillos, E. Wiewiora, and S. Belongie. Objects in context. In *CVPR*, 2007. 1, 2

[20] B. C. Russell, A. Torralba, C. Liu, R. Fergus, and W. T. Freeman. Object recognition by scene alignment. In *NIPS*, 2007. 6

[21] B. C. Russell, A. Torralba, K. P. Murphy, and W. T. Freeman. LabelMe: a database and web-based tool for image annotation. *IJCV*, 77(1-3):157–173, May 2008. 2

[22] E. B. Sudderth, A. Torralba, W. T. Freeman, and A. S. Willsky. Learning hierarchical models of scenes, objects, and parts. In *ICCV*, 2005. 2

[23] A. Torralba. Contextual priming for object detection. *IJCV*, 53:2, 2003. 1, 3, 4

[24] A. Torralba, K. P. Murphy, and W. T. Freeman. Contextual models for object detection using boosted random fields. In *NIPS*, 2005. 1

[25] Z. Tu. Auto-context and its application to high-level vision tasks. In *CVPR*, 2008. 1

[26] J. Winn, A. Criminisi, and T. Minka. Object categorization by learned universal visual dictionary. In *ICCV*, 2005. 2