

mon between viewpoints (e.g., the colour of a car is independent of the viewpoint), which we call *appearance-sharing*. Secondly, boundary appearance potentials which are used at the bottom levels of the hierarchies for different contours.

Part-sharing and *appearance-sharing* (see figure 1) are key ingredients of our approach since they not only yield a compact representation of many object/viewpoints but also make inference and learning more efficient. Inference is done by extending the compositional inference algorithm [19] so that inference is done over the dictionaries and hence is only performed once for each flexible part and is independent of the number of times that the part appears in the object/viewpoint models. Similarly part-sharing and appearance sharing simplify learning by requiring less training data (e.g., if we have already learnt the flexible parts to model a horse then we only need a little more data to learn a model of a cow).

We tested our approach on four challenging public datasets. We obtain strong performance when tested on multi-view multi-object detection. When evaluated on restricted tasks – single object or single viewpoint – we achieve state-of-the-art results when compared to alternative methods. We emphasize that we only used limited training data – by exploiting shape-sharing and appearance-sharing – but succeeded in learning a multi-level dictionary of RCMs which capture the meaningful shape parts shared between different objects and viewpoints.

1.1. Related work

Multi-view object detection can be considered as a special case of multi-object detection, although there is a small difference caused by appearance-sharing. But there have been few attempts to address simultaneous multi-view multi-object detection. Part sharing has been studied, but usually for two-layer structures such as [15] and [12]. We argue that hierarchies with more layers are needed because objects have deformations and shared parts at several ranges of scale. Fidler and Leonardis [8] have used hierarchical structures and part sharing, but they do not use multi-view appearance sharing which is very useful when training data from rare viewpoints is not easy to collect. The hierarchical sharing used in [10] has no latent position variables to model deformation explicitly, and no discriminative appearance learning. Torralba *et al.* [16] have used feature sharing in boosting, but they have not considered multi-view and multi-objects simultaneously. Our work is closest technically to [19] which has used hierarchical models, but this work did not address multi-objects and multi-viewpoints. Bai *et al.* [3] use shared skeleton to model varied poses of objects. It is worth noting that stochastic grammars of Zhu and Mumford [20] offer a principled framework for the representation of images with sharing.

2. RCMs and Families of RCMs

In this section we first describe an RCM model for a single object. Then we describe how we can construct families of RCMs which can simultaneously model multiple appearances and viewpoints and how inference and learning of RCMs can be performed.

2.1. An RCM for a Single Object

An RCM represents an object recursively as a composition of parts. These parts are deformable and are represented in turn by RCMs. More formally, an RCM with root node \mathcal{R} is specified by a graph $(\mathcal{V}, \mathcal{E})$ where \mathcal{V} are the nodes which represent object parts. \mathcal{E} are the graph edges which encode the parent-child relations of parts, see figure (1), where $\nu \in ch(\mu)$ specifies that node ν is a child of node μ . \mathcal{V}_{leaf} denotes the set of leaf nodes. State variables $w_\mu = (\mathbf{x}_\mu, \theta_\mu, s_\mu)$ are defined at each node $\mu \in \mathcal{V}$ of the graph, where $\mathbf{x}_\mu, \theta_\mu, s_\mu$ represent the position, orientation, and size of the parts respectively, $w_{ch(\mu)}$ denotes the states of all the child nodes of μ .

The conditional distribution over the state variables is given by $P(W|\mathbf{I}) = \frac{1}{Z(\mathbf{I})} \exp\{-E(W, \mathbf{I})\}$ where \mathbf{I} refers to the input image, W denotes all state variables, $Z(\mathbf{I})$ is the partition function and $E(W, \mathbf{I})$ is defined as:

$$E(W, \mathbf{I}) = \sum_{\mu \in \mathcal{V}/\mathcal{V}_{leaf}} \psi_\mu(w_\mu, w_{ch(\mu)}) + \sum_{\mu \in \mathcal{V}_{leaf}} \phi_\mu(w_\mu, \mathbf{I}) + \phi_{\mathcal{R}}(w_{\mathcal{R}}, \mathbf{I}) \quad (1)$$

where the shape potentials $\vec{\psi} = \{\psi_\mu(w_\mu, w_{ch(\mu)}) : \mu \in \mathcal{V}/\mathcal{V}_{leaf}\}$ (the black lines in figure (1)) encode the spatial pairwise relationships between the states of the parent nodes and those of their children. These are similar to those used in [19].

The appearance potentials which relate the states of the nodes to the input image are defined at the leaf nodes and the root node. The boundary potentials $\vec{\phi} = \{\phi_\mu(w_\mu, \mathbf{I}) : \mu \in \mathcal{V}_{leaf}\}$ at leaf nodes (the red lines in figure (1)) account for the part of the object appearance which is easily localizable on the corresponding boundary segments. There are 10 oriented generic boundary segments (10 colored panels on the extreme right of figure 1) which can be composed to form varied contours.

The body potential $\phi_{\mathcal{R}}$ at the root node (the blue lines in figure (1)) is defined by $\phi_{\mathcal{R}}(w_{\mathcal{R}}, \mathbf{I}) = (1/|R(\Omega, w_{\mathcal{R}})|) \sum_{\mathbf{x} \in R(\Omega, w_{\mathcal{R}})} \phi(\mathbf{x}, \mathbf{I})$ where $\phi(\mathbf{x}, \mathbf{I})$ describes the non-localizable appearance cues such as the texture/material properties of image patches at locations \mathbf{x} inside the body of the object. The region $R(\Omega, w_{\mathcal{R}})$ of the body is specified by an object mask Ω which can be located, scaled, and rotated by the state $w_{\mathcal{R}}$ of the root node.

We will describe how to learn the appearance potentials in section (4.2).

Hence an RCM can be specified by its graph and potentials: $(\mathcal{V}, \mathcal{E}, \vec{\psi}, \vec{\phi}, \phi_R)$. Figure (1) shows two examples of RCMs for cars seen from two viewpoints. Next we will discuss how the RCMs for multiple objects can be merged to form a compact representation.

2.2. Sharing RCMs

Suppose we want to model a large number of objects seen from different viewpoints. One option is to provide separate RCMs for each object/viewpoint as shown in figure (1), but this is a redundant representation, since it ignores the fact that objects can share parts, and is not efficient computationally. A better strategy is to represent the objects/viewpoints in terms of shared parts and shared appearance properties.

It is reasonable to expect that two objects may share common parts which have same spatial layout even though their bounding contours may differ greatly. For example, in figure (1), the two cars are seen from different viewpoints but have similar roofs (the dotted rectangles) and so can be encoded by the same horizontal segments. This means that the RCMs for the two objects can share an RCM for this part with the same shape potential $\vec{\psi}$. It will only be necessary to keep one copy of this RCM, hence reducing the complexity. Part sharing can also be allowed between different object categories which simplifies the representation for multiple objects and makes inference (parsing) and learning much more efficient. We will describe the mechanisms for part sharing and learning in sections (2.3,4.1).

Sharing can be applied to appearance as well. It is reasonable to assume that the appearance of the body of each object remains fairly similar as the viewpoint varies. In other words, certain image properties encoded by the appearance potentials such as color and image gradients will only depend on the identity of object classes, but not their poses. For example, the RCMs for two cars seen from different viewpoints share the same body appearance $\phi_{\mathcal{R}}$ (blue lines) because they are made from similar materials. Similarly, the appearances of the local boundary segments encoded by $\vec{\phi}$ (red lines) are independent of the global shape of the object boundary. We use one body potential and 10 boundary potentials for one object category, no matter how many viewpoints are considered. The model complexity of RCMs for appearance is linear in size of object classes, but invariant to the number of viewpoints.

Figure (1) shows the probabilistic maps of the body and the boundary which are obtained from the appearance potentials $\phi_{\mathcal{R}}$ and $\vec{\phi}$, respectively. It is clear that the two types of appearance potentials (body and boundary) give strong cues for detecting the body and boundary of the object, despite variations in viewpoint and background. Appearance

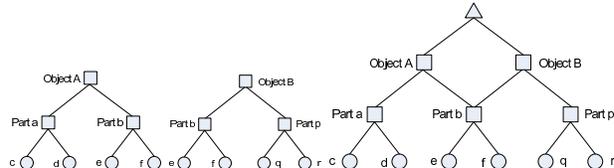


Figure 2. An object A can be represented by an RCM (left) which is specified in terms of parts a, b which are more elementary RCMs. A second object B is also represented by an RCM (middle) but it is more efficient to represent A and B together (right) making their shared part b explicit. The family of objects A, B is specified by the dictionaries $\mathcal{T}^3 = \{A, B\}$, $\mathcal{T}^2 = \{a, b, p\}$, $\mathcal{T}^1 = \{c, d, e, f, q, r\}$.

sharing means that we can learn the body appearance model even though there are only a few training images from each viewpoint. We will describe the details of appearance learning in section (4.2).

2.3. RCM Dictionaries and Shared Parts

We now extend the representation of single graph model described in section 2.1 to a family of graphs (RCMs) for different object/viewpoints. We start with some toy examples illustrated in figure (2). The left and middle panels show three-layer RCMs for two objects A and B . Each object separately can be described in terms of a hierarchical dictionary of the RCMs which are used to compose it – e.g., for object A the dictionary is $\mathcal{T}^3 = \{A\}$, $\mathcal{T}^2 = \{a, b\}$, and $\mathcal{T}^1 = \{c, d, e, f\}$ (where i indexes the level of the RCM). Because objects A and B share a common sub-part b their dictionaries overlap and it is more efficient to represent them by a family of RCMs (right panel) with dictionary $\mathcal{T}^3 = \{A, B\}$, $\mathcal{T}^2 = \{a, b, p\}$, and $\mathcal{T}^1 = \{c, d, e, f, q, r\}$. We will show in sections (3,4) that this family structure makes inference and learning far more efficient.

More formally, we define a hierarchical dictionary $\mathcal{T} = \cup_{l=1}^L \mathcal{T}^l$ where \mathcal{T}^l is the dictionary of RCMs with l levels. A dictionary is of form $\mathcal{T}^l = \{t_a^l : a = 1, \dots, n^l\}$ where each t_a^l is a RCM with l levels, described by a quadruplet $(\mathcal{V}_a^l, \mathcal{E}_a^l, \vec{\psi}_a^l, \vec{\phi}_a^l)$ which define a probabilistic distribution on graph (see equation 1). n^l is the number of RCMs at that level. The dictionaries are constrained, as described above, so that any RCM in \mathcal{T}^l must be a composition of three RCMs in \mathcal{T}^{l-1} . For simplicity, we require that every level- l RCM must be used in at least one level- $(l+1)$ RCM (we can prune the dictionaries to ensure this). The RCMs at the top level L will represent the objects/viewpoints and we call them *object-RCMs*. An object-RCM can be specified by a list of its elements in the sub-dictionaries – $(a^{L-1}, b^{L-1}, c^{L-1}), (a^{L-2}, \dots)$ – together with its potential $\psi(\cdot, \cdot)$ giving the relations of its root node to the roots nodes of $(a^{L-1}, b^{L-1}, c^{L-1})$ and its body potential $\phi_{\mathcal{R}}(\cdot)$.

Shared-RCMs are elements of the dictionaries which are used in different object-RCMs. Note that if an l -level RCM is shared between two, or more, object RCMs then all its sub-RCMs must also be shared between them. Clearly the most efficient representation for the set of object/viewpoints

is to share as many RCMs as possible which also leads to efficient inference algorithms and reduces the amount of training data required.

3. Inference

We can perform efficient inference on a family of RCMs simultaneously. First observe that we can perform inference on a single RCM by exploiting the recursive formulation of the energy defined in equation (1):

$$E(W_\mu, \mathbf{I}) = \psi_\mu(w_\mu, w_{ch(\mu)}) + \sum_{\nu \in ch(\mu)} E(W_\nu, \mathbf{I}). \quad (2)$$

Here W_μ denotes the state of the sub-graph with μ as root node. For the leaf nodes $\nu \in \mathcal{V}_{leaf}$, we define $E(W_\nu, \mathbf{I}) = \phi_\nu(w_\nu, \mathbf{I})$ (for a leaf node $W_\nu = (w_\nu)$). At the root node, we must add the body potential $\phi_{\mathcal{R}}(w_{\mathcal{R}}, \mathbf{I})$ on the right-hand side of equation (2), see section (4.2).

To perform inference for an entire family we work directly on the hierarchical dictionaries. For example in figure (2), when we seek to detect objects A and B , we search for probable instances for the elements $\{c, d, e, f, q, r\}$ of the first level dictionary \mathcal{T}^1 , then compose these to find probable instances for the second level dictionary \mathcal{T}^2 , and then proceed to find instances of the third level \mathcal{T}^3 – which detects the objects. By comparison, performing inference on objects A and B separately, would waste time detecting part b for each object separately. If we only seek to detect one object, then the inference algorithm here reduces to the compositional inference algorithm described in [19].

More precisely, for each RCM t_a^l in dictionary \mathcal{T}^l we find all instances whose energy is below a threshold (chosen to provide an acceptably small false negative rate) and whose states are sufficiently different by selecting the state with locally minimal energy, see [19] for more details of the inference algorithm. Next we form compositions of these to obtain instances of the RCMs from the next level directory \mathcal{T}^2 and proceed in this way until we reach the object-RCMs at the top level. At each stage of the algorithm we exploit the recursive equation (2) to compute the energies for the “parent” RCMs at level l in terms of their “child” RCMs at level $l - 1$. It should be emphasized that this dictionary inference algorithm directly outputs the objects/viewpoints which have been detected as well as their parses – there is no need for a further classification stage.

It is interesting that this inference algorithm incorporates aspects of perceptual grouping in a natural manner. The first few levels of the dictionaries require searching for compositions which are similar to those suggested by Gestalt theorists (and the perceptual grouping literature). But note that unlike standard perceptual grouping rules these compositions are learnt from natural data and arise naturally because of their ability to enable us to detect many objects simultaneously.

The inference described above is performed at different scales of an image pyramid with scaling factor 1.5 (four layers in our experiments). The range of image features (i.e. the resolution of the edge features) are initialized relative to the scales of image pyramid. The inference at one scale of the image pyramid only detects objects within the range of size constrained by the scaling factor.

4. Learning

There are two types of learning involved in this paper. The first type involves inducing the families of RCMs (the dictionaries and shared parts) and learning all the shape potentials $\psi_\mu(w_\mu, w_{ch(\mu)})$. The second type includes learning the object/viewpoint masks Ω , the boundary potentials ϕ and the body potentials $\phi_{\mathcal{R}}$. The first type of learning is the most difficult and requires the use of novel techniques. The second type of learning is standard and relies on the use of standard techniques such as logistic regression.

The input to learning is a set of bounding boxes of different objects/viewpoints and the boundaries of the objects themselves (specified by polygons and line segments). These are all obtained from the LabelMe dataset [11].

4.1. Learning the dictionaries of RCMs

We learn the family of RCMs by learning the sub-dictionaries $\{\mathcal{T}^l\}$ recursively. The input is the set of boundaries of the objects/viewpoints specified by LabelMe [11]. To ensure that we learn re-usable subparts, we proceed on the entire set of boundary examples and throw out the labels which specify what object/viewpoint the boundaries correspond to. At first sight, throwing out this label information seems to make the learning task really difficult because, among other things, we no longer know the number of object-RCMs that we will want at level L (and, of course, we do not know the value of L) but, as we will show, we are able to learn the dictionaries and families automatically including the number of levels.

The algorithm starts by quantizing the orientation of local segments (three pixels) of the boundaries into six orientation bins. We define a dictionary of level-1 RCMs consisting of six elements tuned to these orientations – i.e. \mathcal{T}_0 consists of six models where each model has a single node with state w and a potential $\phi_a(w_a, \mathbf{I})$ where $a = 0, \dots, 5$ and the potential for model a favors (energetically) boundary edges at orientations $a\pi/6$. Then we proceed using the following five steps. *Step 1*: perform inference on the datasets to detect all instances of the level-1 RCMs. *Step 2*: for each triplet a, b, c of RCMs in \mathcal{T}^1 , find all the composite instances $\{w_{\mathcal{R}_a i}\}$, $\{w_{\mathcal{R}_b j}\}$ and $\{w_{\mathcal{R}_c k}\}$ (i, j, k label detected instances of RCMs a, b, c respectively). Reject those compositions which fail a *spatial test for composition* (Two circles, a minimum and a maximum, are draw round the

centers of the three child instances. If all of the child instances lie within the maximum circle of *any* of the child nodes, then the instances are considered close. But they are too close if this happens for the minimum circle). *Step 3*: cluster the remaining compositions in triplet space to obtain a set of prototype triplet clusters. *Step 4*: estimate the parameters – i.e. the potentials $\psi(w_{\mathcal{R}}, (w_{\mathcal{R}_a}, w_{\mathcal{R}_b}, w_{\mathcal{R}_c}))$ – of these triplet clusters to produce a new set of RCMs at level-2 and thereby form a dictionary \mathcal{T}^2 . *Step 5*: prune the dictionary \mathcal{T}^2 to remove RCMs whose instances overlap too much in the images (this requires detecting the instances of these RCMs in the image – performing an overlap test, which puts rectangles enclosing the child RCM instances and evaluates the overlap of these rectangles with those of different RCMs). Output the pruned dictionary \mathcal{T}^2 and repeat the procedure to learn a hierarchy of dictionaries. The procedure automatically terminates when it ceases to find new compositions (step 2).

This algorithm is related to the *hierarchical learning algorithm* which was described in [19]. This algorithm also learnt RCMs by a hierarchical clustering approach which involved learning a hierarchy of sub-dictionaries. But there are some major differences in the applications and techniques since the goal of [19] was to learn a model of a single object from images with cluttered backgrounds. Hence the algorithm relied heavily on the *suspicious coincidence principle* and a top-down stage, neither of which is used in this paper. In this paper, by contrast the goal is to learn multiple object/viewpoints simultaneously while finding reusable parts. Moreover, in this paper the number of object/viewpoints was determined automatically by the algorithm. We did have a discrepancy for one experiment when the algorithm output 119 RCMs at the top level while in fact 120 were present. This discrepancy arose because one object looked very similar from two different viewpoints (in our applications the viewpoints were typically very widely separated).

4.2. The Mask and Appearance Potentials

We now describe how to learn the mask Ω and the appearance potentials $\vec{\phi}$ and $\phi_{\mathcal{R}}$. We know the object/viewpoint labels so the learning is supervised.

The object/viewpoint masks are learnt from the bounding boxes (and object boundaries) specified by LabelMe [11]. This is performed by a simple averaging over the boundaries of different training examples (for fixed object and viewpoint). The body and boundary potentials are learnt using standard logistic regression techniques [1] with image features as input (see below). The boundary potentials are the same for all objects and viewpoints. The body potentials are specific to each object but are independent of viewpoint. To achieve good performance, the appearance potentials $\vec{\phi}$ and $\phi_{\mathcal{R}}$ used in equation 1 are weighted by two

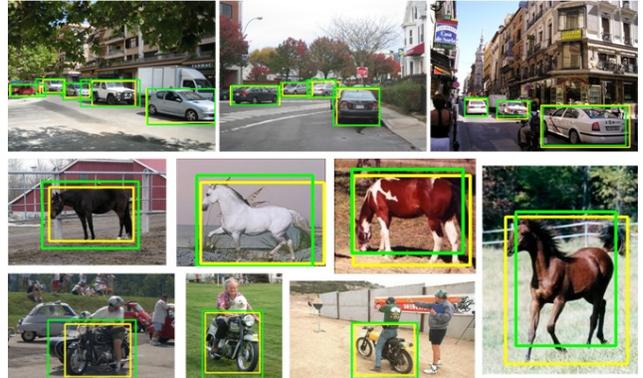


Figure 4. This figure shows the detection results (green boxes) of RCMs on several public datasets, see section (5), compared to groundtruth (yellow boxes). This illustrates that RCMs are able to deal with different viewpoints, appearance variations, shape deformations and partial occlusion.

scalar parameters set by cross validation.

The image features include the greyscale intensity, the color (R,G, B channels), the intensity gradient, Canny edges, the response of DOG (difference of Gaussians) filters at different scales and orientations (see [18] for more details). A total of 55 spatial filters are used to calculate these features (similar to [14]). The scales of these features are small for the boundary and larger for the body (which need greater spatial support to capture texture and material properties). Two types of boundary features were used depending on the quality of the groundtruth in the applications. For single objects, we use features such as edges and corners – see the probabilistic maps calculated by the body and boundary potentials in figure 1. For multiple objects applications we had poorer quality groundtruth and used simpler edge-based potentials similar to [19] based on the log-likelihood ratio of filter responses on and off edges.

5. Experimental Validation

We evaluated RCMs on four challenging public datasets (see detection results in figure 4): (1) 3D motorbike dataset (PASCAL 05 [6]). (2) Weizmann Horse dataset [4]. (3) LabelMe multi-view car dataset (from [11]). (4) LabelMe multi-view multi-object dataset (collected from [11]). Some typical images from these datasets are shown in figure (6). The first three experiments address multi-view single-class object detection while the last one addresses multi-view multi-object detection for 120 different objects/viewpoints. Table 1 summarizes the dataset characteristics (e.g. the numbers of objects/viewpoints and the size of the training and test images). We used a computer with 3.0Ghz cpu and 8GB memory for all experiments. We provide performance comparison with the best previously published results using standard experimental protocols and evaluation criteria. The performance curves were obtained by varying the threshold on the energy (see section 3).

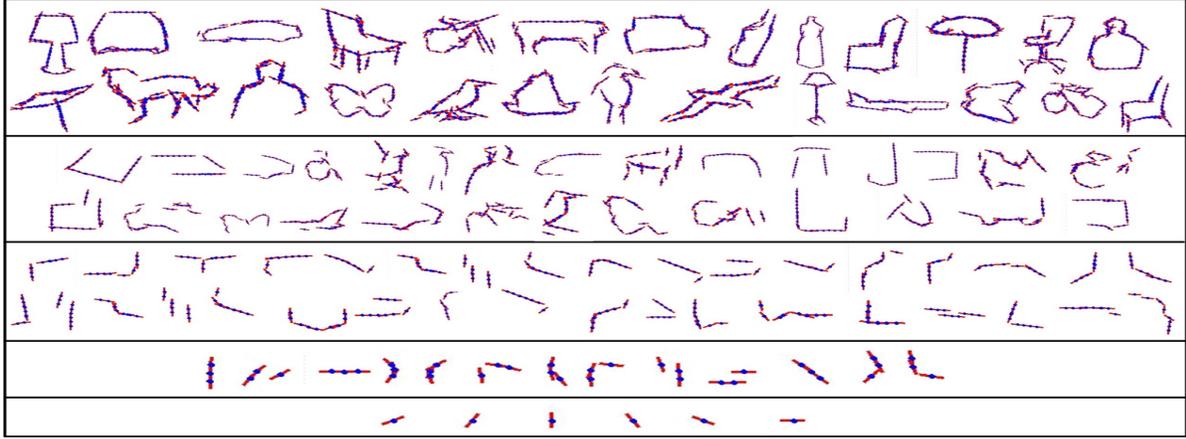


Figure 3. This figure shows the mean shapes of sub-RCMs at different levels. RCMs learn 5-level dictionary from 120 object templates (26 classes).

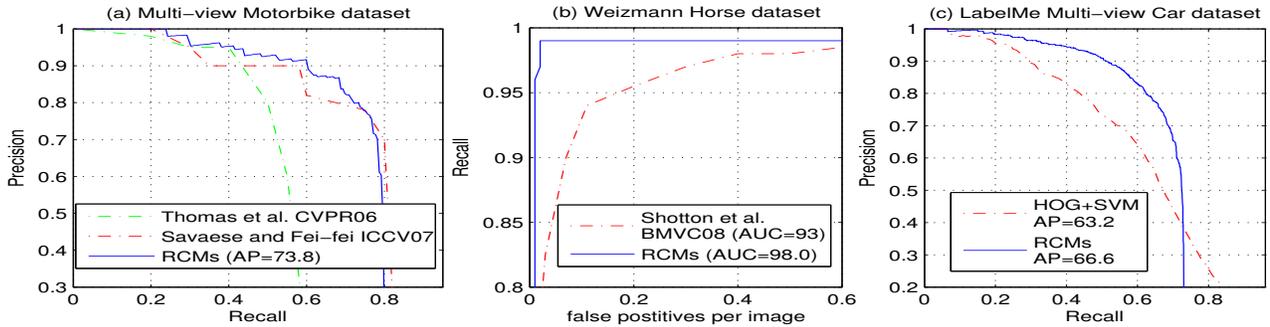


Figure 5. RCMs (blue curves) show competitive performance compared with state-of-the-art methods on several datasets, see section (5).

Datasets	Motorbike	Horse	Car	26 classes
Objects/Viewpoints	16	5	12	120
Training Images	458	100	160	843
Test Images	179	228	800	896

Table 1. The four datasets. The number of viewpoints for each object varies depending on the dataset. All viewpoints are distinct.



Figure 6. Examples of the images used to train RCMs.

5.1. Multi-view Single Object Detection, View Estimation and Shape Matching

Multi-view Motorbike Detection. For this PASCAL dataset we learn RCMs from the training set (12-16 viewpoints) and use the same test images as in [15]. We compare RCMs with the best results reported on this dataset by using precision-recall curves. As can be seen in figure 5(a) RCMs are comparable with [12] and much better than [15] (who do not report the exact average precisions).

Deformable Horse Detection. We applied RCMs (5

viewpoints) to the Weizmann horse dataset which was split into 50, 50, 228 images for training, validation and testing respectively. The Precision-Recall curves in figure 5(b) shows that RCMs have average precision of 98% (almost perfect) on this dataset and outperform the state-of-art method (AP=93%) [13].

Multi-view Car Detection and View estimation. We created a new view-annotated multi-view car dataset by collecting car images from the LabelMe datasets. We learnt RCMs (12 viewpoints) for this dataset and also trained HOG-SVM [5] for comparison. The basic characteristics of the dataset are shown in figure 8 and observe that RCMs require significantly less training data (because of reusable parts). Figure 5(c) plots the precision-recall curves of RCMs and HOG+SVM's evaluated on 800 test images which contain 1729 car instances (an average of 2.2 per image). Observe that the average precision of RCMs is 66.6% which is better than the 63.2% of HOG+SVMs. At the point of (recall=70%), RCMs find 1210 cars together with 0.8 false alarms per image. The processing time of RCMs for detection and viewpoint estimation is about 1.5 minutes per image (of size 1200×900). Figure 7 shows the viewpoint confusion matrix for RCMs. The miss detection rate per view are less than 10%, as shown in the last column in figure 7). The viewpoint is estimated correctly 29.1% of the

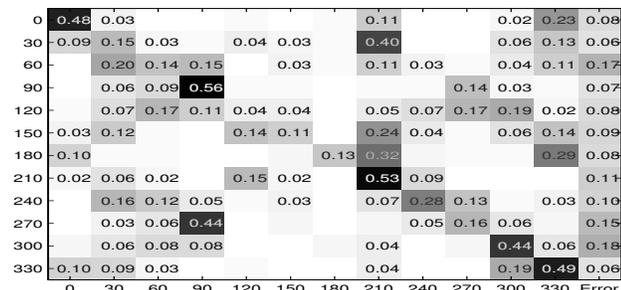


Figure 7. We evaluate the viewpoint classification performance of RCMs for 12 views (with angle range of 30 degrees) by the 12×12 confusion matrix (first 12 columns) and the miss rate of detection for each view (last column).

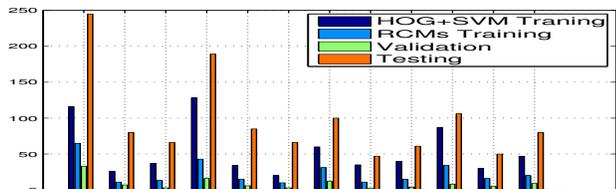


Figure 8. We show the number of training and test images for each viewpoints of cars. Observe that RCMs require much less training data than HOG+SVM. The total numbers of car instances are 660, 284, 108, 1175 for HOG+SVM training, RCMs training, validation, and testing respectively.

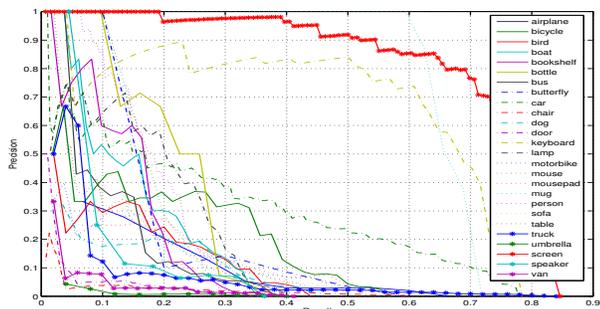


Figure 10. Multi-class precision-recall curves.

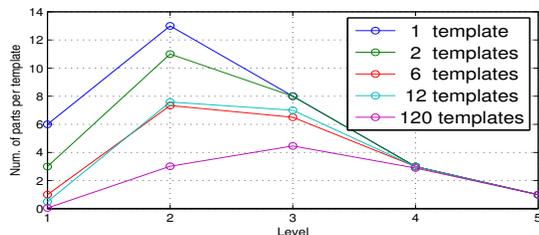


Figure 11. Each curve plots the proportion of parts to object/viewpoints as a function of the levels of the parts (for five families containing different numbers of object/viewpoints). The overall representation cost can be quantified by the size of area under the curves. Observe that the larger the families then the lower the cost and the more the amount of part sharing.

time, but most of the errors are either adjacent viewpoints or symmetric viewpoints (i.e. 180 degrees). By comparing to figure (8) observe that the error rate of each viewpoint is relatively insensitive to the amount of training data.

5.2. Multi-view Multi-Object Detection

For the most challenging task of simultaneous multi-view multi-object detection, we create a new dataset from LabelMe which contains 26 object classes – including

screen, keyboard, mouse, speaker, bottle, mug, chair, can, sofa, lamp – from different viewpoints to give a total of 120 objects/viewpoints.

For this application the most impressive results was our ability to *automatically learn the hierarchical dictionaries including the object models*. Without using the object/viewpoint label our learning procedure output 5 levels where the top level consisted of 119 RCMs, which corresponded to the 120 object/viewpoints (with one model being shared between two different viewpoints). We illustrate the RCMs in figure (3) which shows their mean shapes (we cannot easily display their variations). Observe that level-5 (the top-level) has discovered 119 RCMs for the objects while level-4 has discovered structures which are clearly object specific. By contrast, the low-level (levels 2 and 3) show generic structure – such as oriented straight lines, parallel lines, Y-junctions, and T-junctions – which have often been discussed in the perceptual grouping and gestalt psychology literature (but not learnt in this way from real data).

The results for detecting these different viewpoint/objects from real images is reasonable considering the difficulty of this task but not spectacular, as shown by the precision-recall curves of multiple classes evaluated on the test images in figure (10). It shows proof of concept of our approach but needs to be supplemented by additional appearance cues and, probably discriminative learning, to produce a system with high performance on this task.

To understand how well we are learning to share parts, we compute a matrix which plots the objects/viewpoint against the subparts (RCMs) in the different sub-dictionaries, see figure (9). This shows, not surprisingly, that many RCMs in the lower level dictionaries are shared by all objects/viewpoints but that sharing becomes less frequent for higher level sub-dictionaries. To further illustrate this point we consider RCM families containing different numbers of object/viewpoints. For each family, we plot in figure 11 the proportion of parts used *normalized by the total number of object/viewpoints* as a function of the level of the part. If the family contains only a single object/viewpoint then this simply plots the number of parts at different levels of the hierarchy (e.g. six parts at the bottom level). But as the number of object/viewpoints in the family increases then the proportion of parts to object/viewpoints decreases rapidly for the low level parts. For families with many object/viewpoints, the proportion increases monotonically with the number of levels. All these plots start converging at level four since these high level parts tend to be very object/viewpoint specific.

6. Conclusions

This paper describes a family of recursive compositional models which are applied to multi-view multi-object detection. The decomposition of the appearance and part ap-

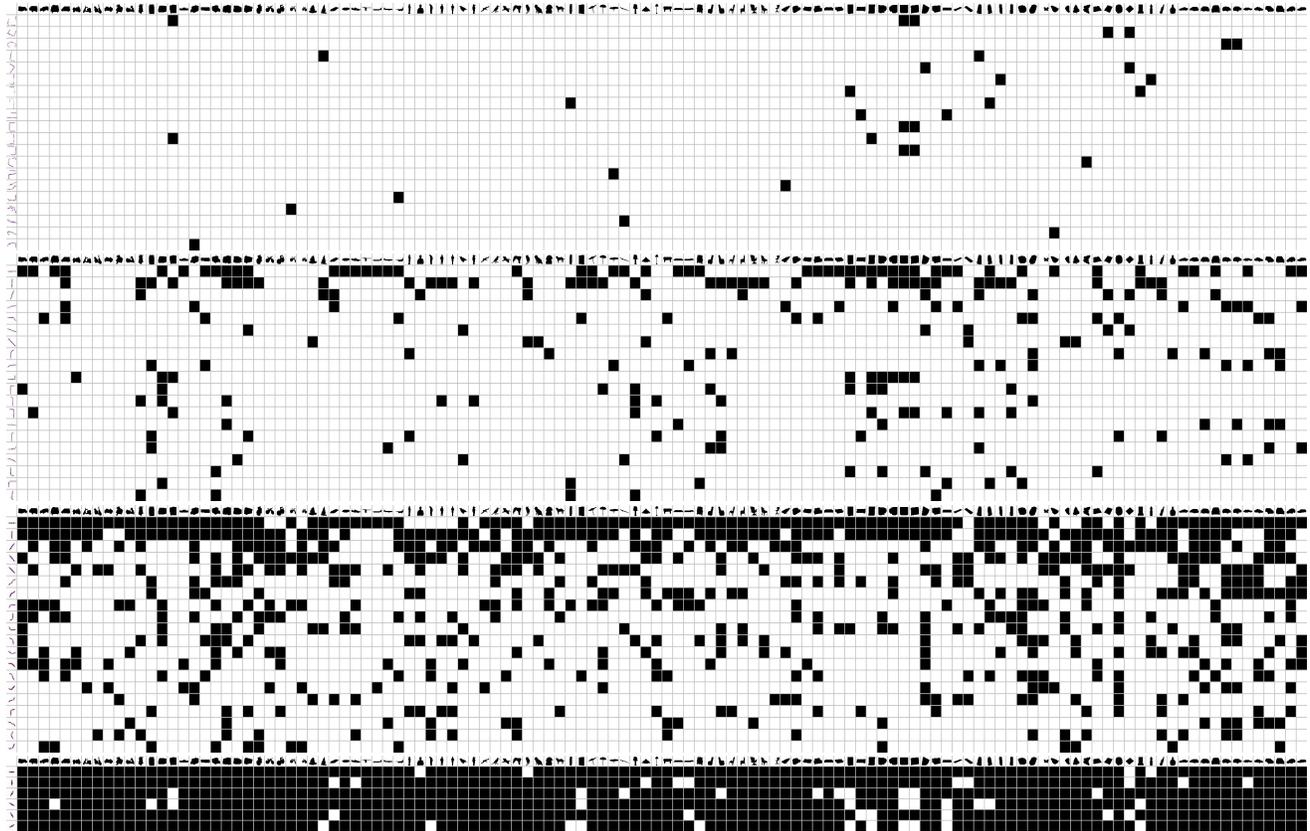


Figure 9. In this part sharing matrix the horizontal axis represents the different object and viewpoint models (120) and the vertical axis represents some of the sub-dictionary subparts at the different levels – level 1 is at the bottom followed by levels 2,3,4. Black squares indicate that a specific sub-part (vertical axis) is used in a specific object/viewpoint model (horizontal axis). Observe that the subparts in the lowest level sub-dictionary are used by almost all the objects/viewpoints. In general, subparts in the lower level sub-dictionaries are frequently shared between objects and viewpoints but this sharing decreases for higher levels where the sub-dictionaries become more object specific.

pearance cues and the use of part-sharing and appearance-sharing enables us to represent a large number of object templates in a compact way. Moreover, part-sharing and appearance-sharing enables efficient learning and inference. We demonstrated the effectiveness of RCMs by the evaluations on four public challenging datasets. Our results show that RCMs achieve state-of-the-art detection performance.

Acknowledgments. Funding for this work was provided by NGA NEGI-1582-04-0004, MURI Grant N00014-06-1-0734, NSF Career award IIS-0747120, AFOSR FA9550-08-1-0489, NSF IIS-0917141 and gifts from Microsoft, Google and Adobe. Thanks to the anonymous reviewers for helpful feedback.

References

- [1] E. L. Allwein, R. E. Schapire, and Y. Singer. Reducing multiclass to binary: A unifying approach for margin classifiers. *JMLR*, 2000. 5
- [2] Y. Amit, D. Geman, and X. Fan. A coarse-to-fine strategy for multiclass shape detection. *IEEE Transactions on PAMI*, 2004. 1
- [3] X. Bai, X. Wang, W. Liu, L. J. Latecki, and Z. Tu. Active skeleton for non-rigid object detection. In *ICCV*, 2009. 2
- [4] E. Borenstein and S. Ullman. Class-specific top-down segmentation. In *ECCV*, 2002. 5
- [5] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *CVPR*, 2005. 6
- [6] M. Everingham. The 2005 pascal visual object class challenge. In *In Selected Proceedings of the First PASCAL Challenges Workshop*, 2005. 5
- [7] P. Felzenszwalb, D. McAllester, and D. Ramanan. A discriminatively trained, multiscale, deformable part model. In *CVPR*, 2008. 1
- [8] S. Fidler and A. Leonardis. Towards scalable representations of object categories: Learning a hierarchy of parts. In *CVPR*, 2007. 2
- [9] D. Hoiem, C. Rother, and J. Winn. 3d layoutcrf for multi-view object class recognition and segmentation. In *CVPR*, 2007. 1
- [10] K. Mikolajczyk, B. Leibe, and B. Schiele. Multiple object class detection with a generative model. In *CVPR*, 2006. 2
- [11] B. Russell, A. Torralba, K. Murphy, and W. T. Freeman. Labelme: a database and web-based tool for image annotation. *IJCV*, 2008. 4, 5
- [12] S. Savarese and L. Fei-Fei. 3d generic object categorization, localization and pose estimation. In *ICCV*, 2007. 1, 2, 6
- [13] J. Shotton, A. Blake, and R. Cipolla. Efficiently combining contour and texture cues for object recognition. In *BMVC*, 2008. 6
- [14] J. Shotton, J. M. Winn, C. Rother, and A. Criminisi. TextonBoost: Joint appearance, shape and context modeling for multi-class object recognition and segmentation. In *ECCV*, 2006. 5
- [15] A. Thomas, V. Ferrari, B. Leibe, T. Tuytelaars, B. Schiele, and L. V. Gool. Towards multi-view object class detection. In *CVPR*, 2006. 1, 2, 6
- [16] A. Torralba, K. P. Murphy, and W. T. Freeman. Sharing visual features for multiclass and multiview object detection. *IEEE Trans. on PAMI*, 2008. 2
- [17] Y. Wu, Z. Si, C. Fleming, and S. Zhu. Deformable template as active basis. In *ICCV*, 2007. 1
- [18] S. Zheng, Z. Tu, and A. Yuille. Detecting object boundaries using low-, mid-, and high-level information. In *CVPR*, 2007. 5
- [19] L. Zhu, C. Lin, H. Huang, Y. Chen, and A. L. Yuille. Unsupervised structure learning: Hierarchical recursive composition, suspicious coincidence and competitive exclusion. In *ECCV*, 2008. 2, 4, 5
- [20] S. Zhu and D. Mumford. A stochastic grammar of images. *Foundations and Trends in Computer Graphics and Vision*, 2006. 2