Combining Regions and Patches for Object Class Localization

Caroline Pantofaru[1] * Gyuri Dorkó[2] Cordelia Schmid[3] Martial Hebert[1]

[1] The Robotics Institute, CMU, USA [2] TU Darmstadt, Germany [3] INRIA Rhône-Alpes, France

crp@ri.cmu.edu, dorko@mis.tu-darmstadt.de, cordelia.schmid@inrialpes.fr, hebert@ri.cmu.edu

Abstract

We introduce a method for object class detection and localization which combines regions generated by image segmentation with local patches. Region-based descriptors can model and match regular textures reliably, but fail on parts of the object which are textureless. They also cannot repeatably identify interest points on their boundaries. By incorporating information from patch-based descriptors near the regions into a new feature, the Region-based Context Feature (RCF), we can address these issues. We apply Region-based Context Features in a semi-supervised learning framework for object detection and localization. This framework produces object-background segmentation masks of deformable objects. Numerical results are presented for pixel-level performance.

1. Introduction

Object class detection and localization has been a frequently studied problem in the computer vision literature. Although the general problem involves detecting and exactly localizing all parts of an object, in many cases the problem is interpreted as a search for a bounding box surrounding the object, with the box generally axis-aligned [1, 22, 24]. Bounding boxes, however, are somewhat unsatisfactory as a final result because they do not capture the true shape of the object. This is especially problematic for deformable objects, wiry objects and objects with occlusions, whose bounding boxes may contain a majority of non-object pixels, such as those in the first row of Fig. 1. A bounding-box based system also has difficulty searching for multiple objects in the same image. Once the system finds one object, it has only two choices in searching for additional objects: ignore pixels within the first bounding box, thereby missing objects which are in different scene layers, or re-process all of the pixels in the bounding box, wasting time and risking classifying the same pixels as belonging to multiple objects. Bounding box-based systems are also often fully supervised, with training objects completely localized using either a segmentation mask or a bounding box.

With this motivation in mind, this paper works towards addressing the problem of pixelwise object class localization. Pixelwise detection and localization involves assigning each pixel in an image to a class label without a predetermined rigid shape assumption on the spatial support of each label. Some such label masks are given in the second row of Fig. 1. We pose this problem in a semi-supervised framework, where only image-level object occurrence labels are available for the training data.

How can an image be represented to facilitate pixelwise labeling? Many popular object recognition methods represent an image with sparse points extracted using an interest point operator [6, 12, 13, 16] and then represent these points using some descriptor such as the SIFT descriptor [13]. Although these sparse detections may be accurate and repeatable, they do not give information about the other pixels on the object which may lie in textureless neighborhoods. Thus the only options for creating a pixel-level mask using sparse points are either to resort to bounding boxes (or some other rigid structure surrounding the interest points) which include background pixels, or to miss most of the object by simply labeling the points themselves. Another popular approach for representing an image is to use patches with predetermined shapes, such as rectangles, and describe their contents as a vector of grayscale values [1]. These representations generally have a larger spatial support, however their predetermined shape can only make inexact estimates of the object extent, generally a set of rectangles covering the object such as in [8]. If these patches are extracted in a sparse manner, they will also exclude parts of the object with low texture.

One approach to remedy the above situation is to use a dense set of pixels [5] instead of those found using an interest point detector. Although a dense set of points can create an accurate object mask, labeling each pixel separately will be noisy and an elaborate smoothing process will be necessary. Instead, we choose to represent an image as a set of regions extracted by unsupervised image segmentation which are homogeneous in color and texture space. This is similar

*Part of this research was performed under the Intelligent Robotics Development Program, a 21st Century Frontier R&D Program funded by the Korean Ministry of Commerce, Industry and Energy. Caroline Pantofaru was partially supported by a grant from the European Community under the Marie-Curie Visitor project.

to the concept of Superpixels [20]. The spatial support for each region is data-dependent, thus a subset of the regions can model the object support precisely. In addition, if the pixels in a region are all given the same label, we remove the pixel-level noise inherent in dense representations.

Performing an initial image segmentation allows us to take advantage of features such as color which are consistent among the member pixels of an object in one image but not between images. These features are unavailable to approaches which seek to cluster pixels in single images and across multiple images at once. By performing segmentation at multiple image scales and grouping pixels in region intersections, we can create a multi-scale environment in which the image is likely to be over-segmented and object pixels are unlikely to be grouped with background pixels.

One way to describe regions is by the mode of their constituent pixels' textures. A score reflecting how well each texture predicts the object class can be learned and used to classify new regions. This works well for objects with distinctive textures, such as the cheetah in the first column of Fig. 1. We run into problems, however, in trying to classify regions which do not have regular textures which can discriminate between the class and the background. The bicycle's frame has no texture at all and is better encoded by its angles, while the texture on the cheetah's face is not regular and will be broken into different regions. Clearly we need to encapsulate more than the information internal to each region. One possible approach is to use the information from neighboring regions, but this leaves us with two important problems. First, for some objects, key features occur only at the borders between regions, such as the corner of an eye. Second, region shapes (and hence spatial relationships between them) are notoriously unstable. This is where descriptors defined on predetermined pixel supports (patches) are useful; they can find and encode non-repetitive features and their spatial extent is reliable. Thus the major contribution of this work is the incorporation of local patch-based features into a region-based approach through the creation of Region-based Context Features (RCF), and their combination with more common texture-based features.

At a high level, a Region-based Context Feature is a histogram of the (quantized) local descriptors near a region, with proximity defined by the scale of the local descriptor patches. In this manner we can remain within a region-based framework but take advantage of the extra features found by patches, with their location and scale stability. By combining Region-based Context Features with texture-based features in a framework which allows data-dependent feature selection, we can model an object class by the set of features which describe it best, and provide the desired pixel-level object mask. Since explicit angle relationships between regions and points are not modeled, we can find objects which are extremely deformable. By presenting our method in a semi-supervised learning framework in which only image-level labels are necessary, we decrease the amount of work needed for training.

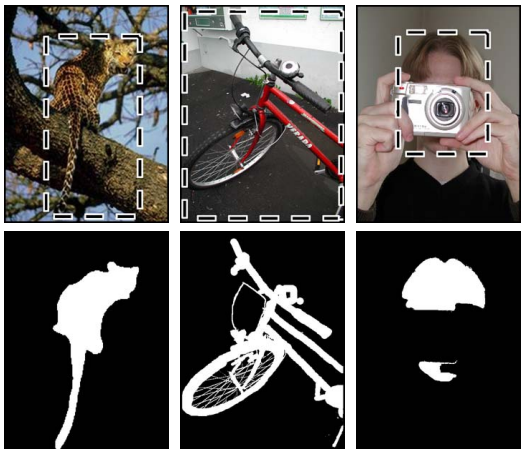


Figure 1. The first row shows deformable objects, wiry objects and objects with occlusions which are not satisfactorily delineated by bounding boxes. The second row shows pixel-level masks of these same objects, which capture their exact shape.

There have been a limited number of attempts to use segmentation for detection or to combine segmentation with other cues. Tu *et al.* [23] combine segmentation with features (such as splines) based on object specific knowledge to detect faces and text segments on natural images. Their DDMCMC-based (Data-Driven Markov Chain Monte Carlo) framework is computationally expensive, however, and requires prior knowledge. Malik *et al.* [14] use local information such as edges to help direct segmentation, but did not perform object detection. On the other hand, the Blobworld system [2] employs segmentation information to retrieve images from a database, however, they do not use local pixel information, nor identify the objects. Opelt *et al.* [19] use interest points [16], similarity measure segmentation, and mean shift segmentation together in an Adaboost framework. They performed object recognition and verified the center of the object by a relatively weak criterion. Many of the described localization methods use fully supervised data [1, 22, 24], sometimes including full segmentation masks [10] or textureless backgrounds [11]. Yu and Shi [26] localize only known and previously seen objects and do not generalize to object classes. Localization is usually evaluated by bounding boxes, or the location of the center. The above methods either did not provide pixel-level localization masks, or did not quantify the masks' accuracy. Kumar *et al.* [7] perform detection and segmentation using pictorial structures and MRFs, however they cannot cope with occlusion or alternate viewpoints.

2. Segmentation and Texture Descriptors

The foundation for our object localization procedure is unsupervised image segmentation. To define the regions in an image, we use mean shift-based image segmentation [3] on position, color and texture descriptors. The texture de-

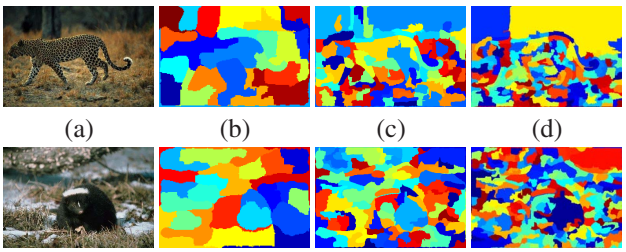


Figure 2. Example segmentations at three image sizes for images from the positive and negative Spotted Cats dataset. Images (a) and (e) are the original images, (b) and (f) are the segmentations of the smallest image size (rescaled to the original size), (c) and (g) are of the middle image size, and (d) and (h) are of the largest image size. Segmentation parameters are kept constant throughout.

scriptors are textons [14], have a feature length of 30 and are extracted using the publicly available implementation from the Berkeley segmentation database website [15]. The color features are computed in the L*u*v* space, and are smoothed over the same 15x15 window used to compute the texton features. In total, we perform segmentation in a 35-dimensional space.

We chose mean shift-based segmentation since it does not require as input the total number of regions, nor does it necessarily create regions of equal size. The only parameters are the kernel bandwidths (radii) for the pixel position, the color, and the texture. A multi-scale representation is achieved by performing image segmentation at three image scales, assigning three different regions to each pixel.

Notice that this procedure extracts regions regardless of their rotation, position, or spatial relationships, and therefore forms possible object regions without a low-level spatial model. Examples of mean shift segmentations on positive and negative images from the Spotted Cats dataset are presented in Fig. 2.

Once an image is segmented at one scale, each region is described by the texture part of its mode, a 30-dimensional vector t. We perform k-means clustering on the texture descriptors from all of the training images at all (3) scales to derive a vocabulary of texture words $T = \{T_i\}_{i=1}^{N_T}$, and assign each region to its nearest neighbor texture word.

3. Region-based Context Features (RCF)

There has been a significant amount of work done on the inclusion of shape (geometric), spatial and context information into object models. Shape information relates the various parts of the object to each other, and has been explored through the concepts of Shape Context [17] and geometric relations in part-based models [25]. Spatial and context models may also relate the object to the other parts of the scene, be they local or global, for example [18]. In our problem, the objects may be highly deformable or seen from highly varying angles, so it is beneficial not to limit the model to strict shape or spatial relationships which enforce a topology among the parts. However, it is useful to model a notion of proximity among object features. This allows us to identify regions which may not have a discriminative texture themselves, but which have close proximity to object features, such as those in Fig. 5. These regions do not have a discriminative texture (in fact they have no texture at all), but they do lie near the dots which line the edge of the butterfly's wing. For this purpose we introduce the concept of Region-based Context Features (RCF).

Consider the segmentation-derived regions in an image. We would like to model the relationships between each region and its surroundings. Although the regions do contain a homogeneous set of pixels (in color and texture space), their shape and size is not repeatable across instances of the same class. In fact, fairly minor variations in lighting or pose can change the shape of the regions, so we wish to rely as little as possible on their shape. This can be accomplished by ignoring the relationships between the regions directly, and instead relying on region-based histograms of local descriptors computed from independent locations and scales in the image.

For our local descriptor we use the 128-dimensional SIFT descriptor [13]. The locations where the descriptors are computed may be sparse, as determined by a local interest point operator and scale selection, or densely arranged over the image and in scale space. Let the set of points in one image be $P = \{p_i\}_{i=1}^{N_P}$, with scales $\{\sigma_i\}$ and local descriptors $\{d_i\}$. Clustering the set of descriptors from all of the training images produces a vocabulary of local descriptor words W of size N_W. Let w_i be the nearest neighbor word to descriptor d_i. We use the scales $\{\sigma_i\}$ of the points to define proximity to regions. The idea is to create a histogram for each region of the local words which are at most $k\sigma_i$ pixels away. These histograms are appended, weighted inversely proportionally to their k values, to create a set of Region-based Context Histograms (RCH).

More specifically, let R be a region in the image, with member pixels $\{r_j\}_{j=1}^{N_R}$. Let h_k be the k-histogram for the region R with N_W bins, one for each word. We build the histograms as follows:

$$h_k(w) = |\{w_i \mid w = w_i, (k-1)\sigma_i < \min_{r_j}\{d(r_j, p_i)\} \leq k\sigma_i\}|. \quad (1)$$

where $d(*,*)$ is the Euclidean distance between the pixel locations. In our implementation, $k \in \{1, 2\}$ and the histogram for $k = 1$ includes the points with distance $k = 0$ (points which are within the region itself). Each h_k is then normalized. For larger k, the histogram contains points which are farther away from, and less related to, the region, and accumulates them over an area which grows quadratically (while k grows linearly). So the h_k are weighted inversely proportionally to k. For our experiments, we use weights of 0.5^k. The (weighted) $\{h_k\}$ are concatenated to get a final feature $H = [h_1, h_2, \ldots, h_K]$. The method for

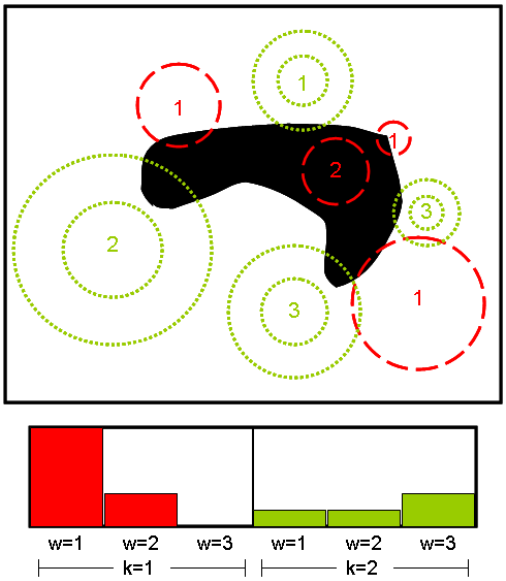


Figure 3. Example of a Region-based Context Histogram (RCH). The top figure is the image, with the black blob representing one region from the segmentation. The red circles represent points p_i in the image that are within $1\sigma_i$ of the region, and the green concentric circles represent points within $2\sigma_i$. The numbers in the circles are the local descriptor words for each point. The bottom figure is the resulting RCH. The left half represents the points that are within $1\sigma_i$ of the region, and the right half $2\sigma_i$. Each entry in the histogram corresponds to a (w,k) pair, where w is a word and k is the multiple of σ. In practice, each k-section of the histogram is normalized and weighted inversely to k.

building an RCH is summarized in Fig. 3.

A vocabulary is created by clustering (using k-means) the RCHs from all of the training images at all the image scales. The Region-based Context Features (RCFs) are the cluster centers, and regions are assigned the RCF which is the nearest-neighbor to their RCH.

4. Computing feature scores

Certain features in the texture and RCF vocabularies will be able to discriminate between an object class and its background. Each feature's discriminative score can be evaluated under a semi-supervised learning model in which images are labeled as positive if they contain the object class, or negative otherwise. No object localization or configuration information is given. Let $P(F_i|O)$ be the conditional probability that a descriptor from an object image is assigned to feature cluster i, and define $P(F_i|\bar{O})$ similarly for non-object images. Combining the texture features and RCFs into one large set, and given the positive and negative labels on the training images, we compute the discriminative power of a given feature F_i as:

$$\tilde{R}(F_i) = \frac{P(F_i|O)}{P(F_i|O) + P(F_i|\bar{O})}$$

Figure 4. Examples of image regions within discriminative texture clusters. The feature on the left is one of the top (best ranked) Spotted Cat texture features, while the one on the right is one of the top Machaon Butterfly texture features.

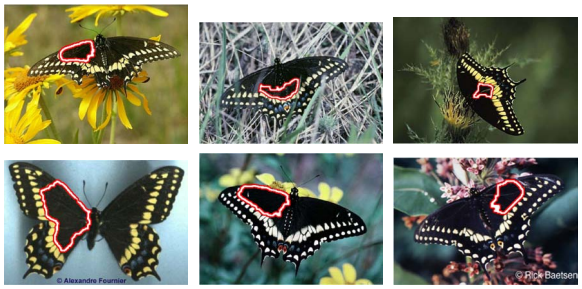


Figure 5. Examples of image regions within a discriminative RCF cluster for black swallowtail butterflies. The red and white outlines denote the regions. Note that these regions would not be discriminative based on texture alone.

This score ranks the features in the same order as the likelihood ratio:

$$R(F_i) = \frac{P(F_i|O)}{P(F_i|\bar{O})}$$

which is the likelihood criterion presented in [4, 21] to select discriminative interest point clusters. Examples of some discriminative feature clusters are given in Fig. 4 and Fig. 5. Notice that the discriminative scores of the texture features and the RCFs are directly comparable.

We can now assign two scores to each region corresponding to the discriminative values of its texture feature and RCF, hence we are ready to localize objects in test images. Note that this entire learning scheme can also be applied in a fully supervised setting by simply replacing positive and negative images with positive and negative regions.

5. One-vs-all Classification

For each new image, we perform unsupervised segmentation and local patch extraction as described above, and compute the nearest neighbors in the texture, T_i, and RCF, RCF_i, vocabularies for each resulting region. We can then determine two values for the region $\tilde{R}(RCF_i)$ and $\tilde{R}(T_i)$. Since these values are on the same scale, we can combine them to get one region score $S_i = \tilde{R}(RCF_i)\tilde{R}(T_i)$. In this formulation, if the texture features cannot discriminate between a certain class and its background, they will all be near 0.5 and will have little to no affect on the region score ordering imposed by the RCFs, and vice versa. If, on the other hand, both texture features and RCFs have a wide range of scores, they can reinforce each other when both

agree or cancel each other out when they disagree. Thus both feature sets can co-exist in one model.

Let m_s be the likelihood map for an image at scale s in which each pixel is assigned its corresponding region's S_i. Then the final one-vs-all pixel map of object class membership likelihood is $M = \prod_{s=1}^{N_s} m_s$. In our experiments the number of scales is $N_s = 3$. Note that if $\{r_s\}$ is a set of regions, one region at each scale, then the pixels in the intersection of the regions, $p \in \bigcap_{s=1}^{N_s} r_s$, all have the same final score. This is reasonable because pixels in the same region at all scales have a common texture and color at all scales. By assigning them the same score we achieve our original goal of classifying similar pixels in a consistent manner and avoiding pixel-level noise.

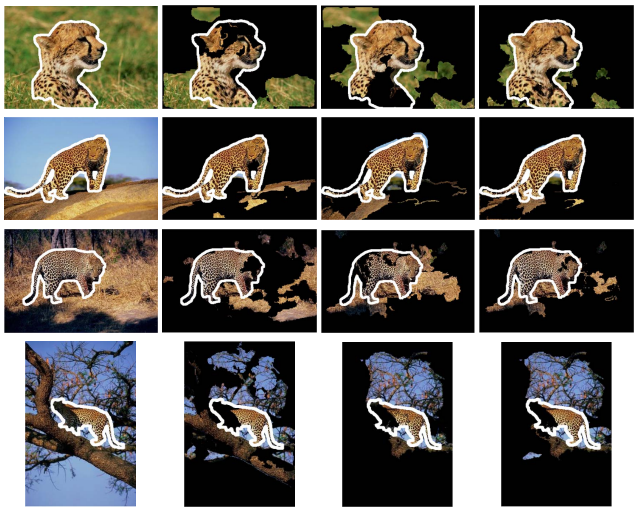
We have described a method for creating one-vs-all classifiers for a dataset. We can extend this to a multi-class classifier by running each of our one-vs-all classifiers over an image and assigning to each pixel the class corresponding to its maximum score, thresholding the responses to identify the background.

6. Experiments

6.1. Spotted Cats

Our first set of experiments uses data from the Corel Image Database, with the folder 'Cheetahs, Leopards and Jaguars' as a positive class and 'Backyard Wildlife' as a negative class. In this paper, this dataset is refered to as the 'Spotted Cats' dataset. Each folder contains 100 images; for training we used 51 positive images (evenly divided among the 3 types of cats) and 50 negative images, with the remainder for testing. All of the images are 640x480 pixels, in either portrait or landscape orientations. However, the size of the actual cats varies considerably, as does the pose, level of occlusion, background and lighting. Many of the images contain multiple cats. All experiments were done using semi-supervised training only. Local patch features were extracted densely, using a regular grid with points spaced 8 pixels apart, and scales of 2, 4, and 8 at each point.

Fig. 6 gives example images and results for this dataset. The first column contains the original images with hand-drawn ground truth for reference (although it was not used for training). The second column contains results obtained using texture-based region features only, in other words $S_i = \tilde{R}(T_i)$. The third column contains results obtained using RCFs only such that $S_i = \tilde{R}(RCF_i)$. Finally, the fourth column contains the results of combining the feature sets, $S_i = \tilde{R}(RCF_i)\tilde{R}(T_i)$. For display purposes, detection thresholds were chosen to give equal error rates on the images. Notice that in the first and third rows, the texture-only classification misses parts of the cats' heads, while in the second example it misses the shadowed section under the chin. These are all low-texture regions. The RCF-only classification is able to properly label these regions, but has trouble with the tails which are mainly surrounded by background. The combined results manage to capture most of the cat silhouettes, and reduce the amount of background noise present. The fourth row shows a poor result where background texture is too dominant and a large part of the object is too shadowed for proper classification.



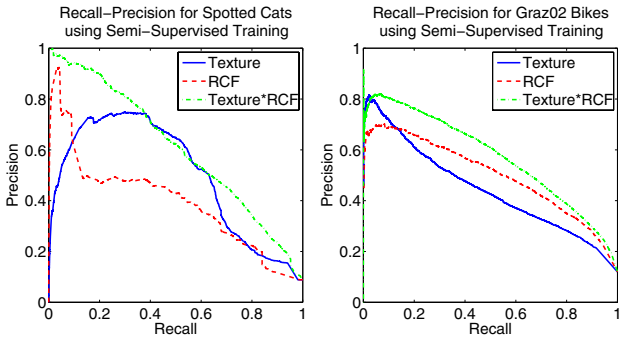
(a) (b) (c) (d)

Figure 6. Sample results for the Spotted Cats dataset. Column (a) contains the original images, column (b) contains results from texture-only classification, column (c) contains results from RCF-only classification, and column (d) contains results from combined texture and RCF classification. The first three rows show good results, while the fourth result is poor. The white outlines are ground truth.

Fig. 7(a) plots recall-precision results for pixel-level classification on the Spotted Cats dataset. Notice that each of the baseline texture-only and RCF-only methods have strengths and weakness. The texture-only method performs poorly at low recall values, however its performance degrades slowly as the recall increases. On the other hand, the RCF-only method has excellent precision at low recall, but the curve drops quickly as the recall increases. By combining the two feature sets we produce a curve which has both high precision at low recall rates, and drops off slowly. Our results include all of the pixels in the image, including those near the borders of the image and the object outline. Due to inevitable inaccuracies in the hand-labeled data, obtaining perfect recall and precision is impossible. Note that previous publications have given only example images as results, they have not given pixel-level numerical results.

6.2. Graz02 Bikes

Our second set of experiments was run on the publicly available 'Graz02' dataset used in Opelt et al [19]. We used the 'Bikes' class as the positive class, and the 'Background' class as the negative class. The data was split in accordance with the original paper, with the odd numbered images in each class for training and the even numbered images for



(a) (b)

Figure 7. Recall-Precision curves. Plot (a) shows results for the Spotted Cats dataset, and plot (b) for the Graz02 Bikes vs Background dataset. Both classifiers were created using semi-supervised training. The blue solid lines represents classifying using region texture only, the red dashed lines using RCFs only, and the green dash-dotted lines using the full region texture and RCF model. Results are for pixel-level classification.

testing. All of the images in this dataset are 640x480 pixels, but the size of the objects varies drastically, as do the pose, level of occlusion, background and lighting. Fig. 7(b) shows the pixel-level performance of the classification. Notice that the combination of texture features and RCFs performs better than either method alone at almost all operating points.

Since our task of pixelwise classification, and the resulting metric of pixelwise recall-precision, differ from previous work, we adjusted our algorithm to output object centers and compared localization performance to Opelt et al [19] using their framework. For the positive (bike) images, we used as our localization the center of the largest connected component of values greater than 0.6 in the likelihood map M. Setting this threshold is equivalent to setting the number of clusters k in [19], which is also set by hand. Since this form of localization is not the main focus of our work, we did not perform a thorough search for the optimal threshold. Using the comparison method from [1] used by Opelt et al., a localization is considered correct if it falls within the ellipse inscribed in the bounding box surrounding the ground truth. In the case of multiple ground truth objects, finding any one is considered correct. Within this framework, we are able to localize 131/150 of the bikes, compared to 115/150 for Opelt et al.

6.3. Butterflies

Our third set of experiments was run on the publicly available 'Butterflies' dataset used in [9]. The dataset consists of 619 images of 7 classes of butterflies: Zebra, Machaon, Monarch (closed position), Monarch (open position), Peacock, Black Swallowtail, and Admiral. The pictures were collected from the internet and hence vary widely in size, quality, focus, object size, object rotation, number of butterflies, lighting, etc. The first 26 images from each butterfly class were used for training, and the remainder for testing. Butterflies themselves are not extremely deformable, however the variety in the images coupled with the similarity between the different butterfly classes makes this a very difficult dataset for pixel-level work. The one-vs-all classifiers in this dataset are required to discriminate between butterfly classes, in addition to discriminating between butterflies and backgrounds. Note that the Monarch (closed position) and Monarch (open position) classes actually contain the same butterfly type, simply in different poses. The Admiral and Peacock classes have minimal texture, making them especially difficult to learn.

Fig. 8 shows example results for each of the butterfly types, classified with one-vs-all classifiers. The columns represent, from left to right, the original images, classification using only texture features, classification using only RCFs, and classification using the combined texture and RCF model. Notice that the texture features are able to extract most of the regular textures on the butterflies, while the RCFs are able to extract some of the nearby texture-less regions. Also, the two models work together to lower background noise. The Zebra butterfly class (first row) presents a fairly easy problem, with all of its body covered by a distinctive texture. The Peacock, Black Swallowtail and Admiral classes (last 3 rows), however, are extremely difficult as the majority of their surface is textureless, with the Peacock and Admiral butterflies having no distinctive regular textures at all. Additionally, silhouette shape is not a strong cue for discriminating between butterfly classes.

Fig. 9 gives the pixel-level recall-precision plots for each of the one-vs-all classifiers, in the same order as the examples in Fig. 8. These datasets demonstrate the performance range of our algorithm. The Zebra classifier produces excellent results due to its regular texture. The combination of textures on the Machaon, Monarch (Closed) and Monarch (Open) give promising results as well, with the combined model improving performance. The Peacock and Admiral classes contain much larger challenges and strain the system since they both lack a distinctive regular texture and contain mainly large uniform regions. In these cases the texture-based classifier is essentially useless, severely handicapping the system.

Finally, we present results for multi-class classification on the butterflies dataset. If the maximum one-vs-all classifier value at pixel p_i is $c^* = max_{c=1}^{C} M_c(p_i)$, then the class at each pixel p_i is:

$$C(p_i) = \begin{cases} c^* & M_{c^*}(p_i) > 1.5(1 - M_{c^*}(p_i)) \\ background & otherwise \end{cases}$$

In Table 10 we can see the pixelwise classification rates for the butterfly classes. The only two low scores came from the Swallowtail and Peacock classes which performed poorly in the one-vs-all tasks as well. Fig. 11 shows some examples of classifications. The multiclass framework seems to have improved classification. One possible explanation for this

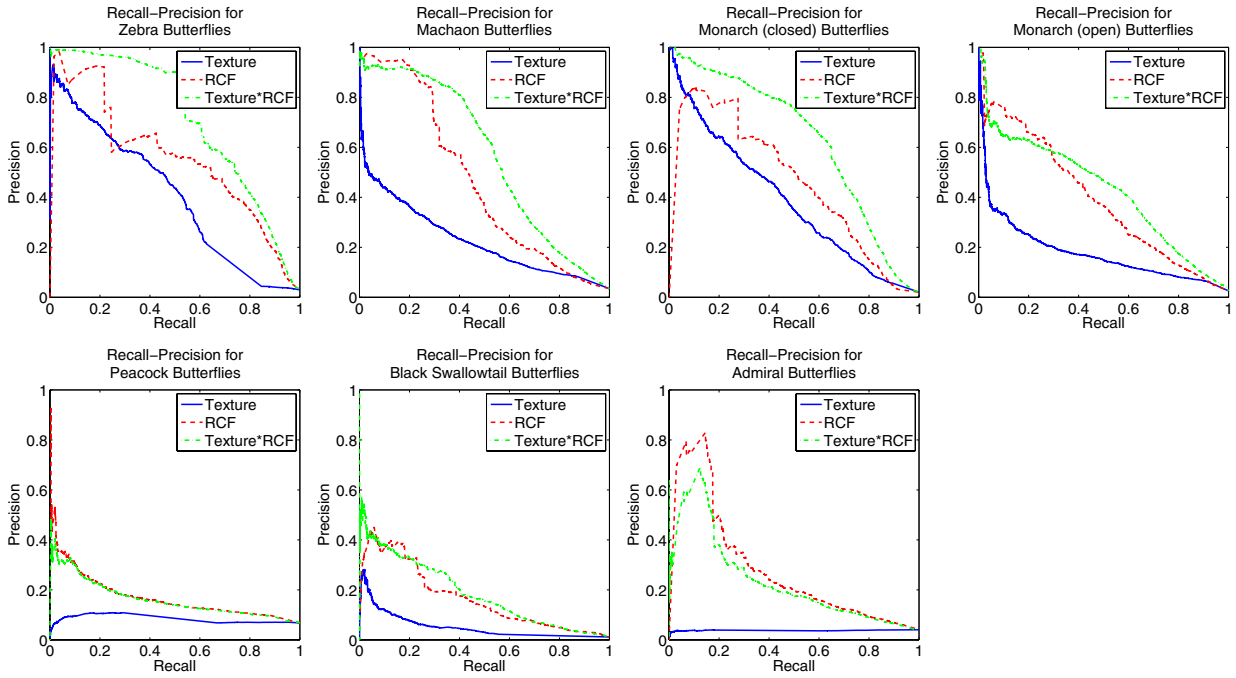


Figure 9. Recall-Precision curve for the one-vs-all classifiers in the Butterflies dataset. All classifiers were created using semi-supervised training. The blue solid line represents classifying using region texture only, the red dashed line using RCFs only, and the green dash-dotted line using the full region texture and RCF model. Results are for pixel-level classification.

is that a concensus between 7 classifiers is required to label a region as background, increasing the precision.

7. Conclusions and future work

In this paper, we have presented a method for performing pixel-level object classification and localization in a region-based framework which incorporates both texture features and a new feature, the Region-based Context Feature. In addition, training is performed in a semi-supervised manner, minimizing the amount of work required. Our framework takes advantage of the deformable nature of segmentation regions to clearly delineate the boundaries of an object and provide a pixel-level mask which is more accurate than a bounding box or other rigid structure. Through the use of texture information and flexible spatial information as encapsulated in the Region-based Context Feature, we can model objects which are highly deformable and contain repetitive patterns, a challenge for interest-point based systems with highly constrained shape models. We have shown that the system performs well on objects which are at least partially covered by repetitive textures, and it is able to model low texture regions by relating to these repetitive textures and shape cues. Objects which have neither strong shape nor texture features are probably better served by other methods. As future work, it would be interesting to try to learn thresholds for the detection task while still only using semi-supervised training data. We would also like to explore methods for identifying the number of objects in an image. Additional filtering would also help to remove some of the spurious results.

References

[1] S. Agarwal, A. Awan, and D. Roth. Learning to detect objects in images via a sparse, part-based representation. *PAMI*, 2004.

[2] C. Carson, S. Belongie, H. Greenspan, and J. Malik. Blobworld: Color- and texture-based image segmentation using em and its application to image querying and classification. IEEE *PAMI*, 24(8):1026–1038, 2002.

[3] D. Comaniciu and P. Meer. Mean shift: A robust approach toward feature space analysis. *PAMI*, 2002.

[4] G. Dorkó and C. Schmid. Selection of scale-invariant parts for object class recognition. In *ICCV*, volume 1, pages 634–640, 2003.

[5] F. Jurie and B. Triggs. Creating efficient codebooks for visual recognition. In *ICCV*, 2005.

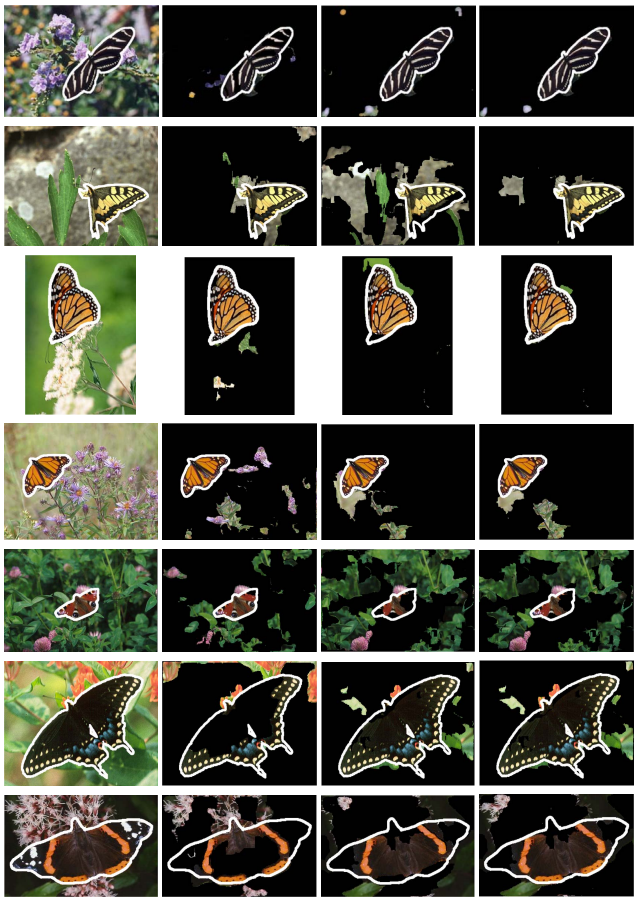
[6] T. Kadir, A. Zisserman, and M. Brady. An affine invariant salient region detector. In *ECCV*, 2004.

[7] M. Kumar, P. Torr, and A. Zisserman. Obj cut. In *CVPR*, 2005.

[8] S. Kumar and M. Hebert. Discriminative fields for modeling spatial dependencies in natural images. In *NIPS*, 2004.

[9] S. Lazebnik, C. Schmid, and J. Ponce. Semi-local affine parts for object recognition. In *BMVC*, 2004.

[10] B. Leibe and B. Schiele. Scale invariant object categorization using a scale-adaptive mean-shift search. In *DAGM'04 Annual Pattern Recognition Symposium*, 2004.



(a) (b) (c) (d)

Figure 8. Sample results for each class within the Butterflies dataset. Column (a) contains the original images, column (b) contains results from texture-only classification, column (c) contains results from RCF-only classification, and column (d) contains results from combined texture and RCF classification. The white outlines are ground truth. The rows present the classes in the same order as the plots in Fig. 9: Zebra, Machaon, Monarch (Closed), Monarch (Open), Peacock, Black Swallowtail, and Admiral. In many cases the texture and RCF representations are complimentary; the texture representation finds regular textures while the RCF representation fills in textureless regions which are close to textured regions. However, in some cases such as in row 5, the classification fails due to very small scale, low object texture, and high intraclass variability.

Butterfly	Classif.	Butterfly	Classif.
Admiral	76.09%	Swallowtail	13.57%
Machaon	53.58%	Zebra	74.94%
Monarch-closed	92.67%	Pixel avg	53.84%
Monarch-open	66.85%	Class avg	57.59%
Peacock	25.63%		

Figure 10. Pixel-level multi-class classification rates for the Butterflies dataset.

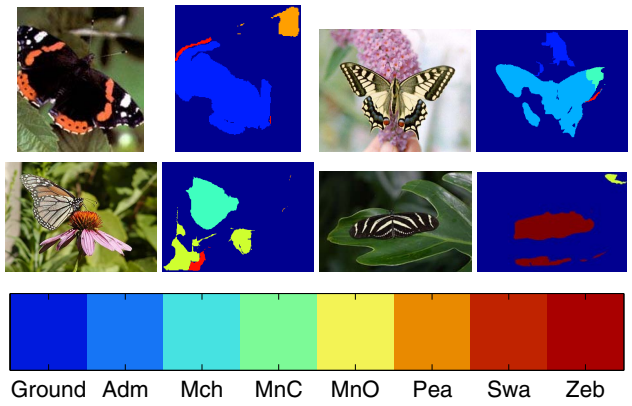


Figure 11. Examples of multi-class classification on the butterflies dataset. Each pair of images shows the original image input and the classified output. Each color in the output represents a butterfly class, with dark blue the background, as given in the color chart.

[11] B. Liebe and B. Schiele. Interleaved object categorization and segmentation. In *BMVC*, 2003.

[12] T. Lindeberg and J. Garding. Shape-adapted smoothing in estimation of 3D depth cues from affine distortions of local 2D brightness structure. In *ECCV*, pages 389–400, 1994.

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

[14] J. Malik, S. Belongie, J. Shi, and T. Leung. Textons, contours and regions: Cue combination in image segmentation. In *ICCV*, 1999.

[15] D. Martin, C. Fowlkes, D. Tal, and J. Malik. A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. In *ICCV*, 2001.

[16] K. Mikolajczyk and C. Schmid. Scale and affine invariant interest point detectors. *IJCV*, 60(1):63–86, 2004.

[17] G. Mori and J. Malik. Estimating human body configurations using shape context matching. *Workshop on Models versus Exemplars in Computer Vision, CVPR*, 2001.

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

[19] A. Opelt and A. Pinz. Object localization with boosting and weak supervision for generic object recognition. In *SCIA*, 2005.

[20] X. Ren and J. Malik. Learning a classification model for segmentation. In *ICCV*, 2003.

[21] C. Schmid. Constructing models for content-based image retrieval. In *CVPR*, 2001.

[22] H. Schneiderman and T. Kanade. A statistical method for 3D object detection applied to faces and cars. In *CVPR*, 2000.

[23] Z. Tu, Z. Chen, A. L. Yuille, and S.-C. Zhu. Image parsing: Unifying segmentation, detection, and recognition. *IJCV*, 2005.

[24] P. Viola, M. J. Jones, and D. Snow. Detecting pedestrians using patterns of motion and appearance. In *ICCV*, 2003.

[25] M. Weber, W. Einhuser, M. Welling, and P. Perona. Viewpoint-invariant learning and detection of human heads. In *Automatic Face and Gesture Recognition*, 2000.

[26] S. Yu and J. Shi. Object-specific figure-ground segregation. In *CVPR*, 2003.