# What Has My Classifier Learned? Visualizing the Classification Rules of Bag-of-Feature Model by Support Region Detection

Lingqiao Liu
CECS, Australian National University
ACT 0200, Canberra, Australia
lingqiao.liu@cecs.anu.edu.au

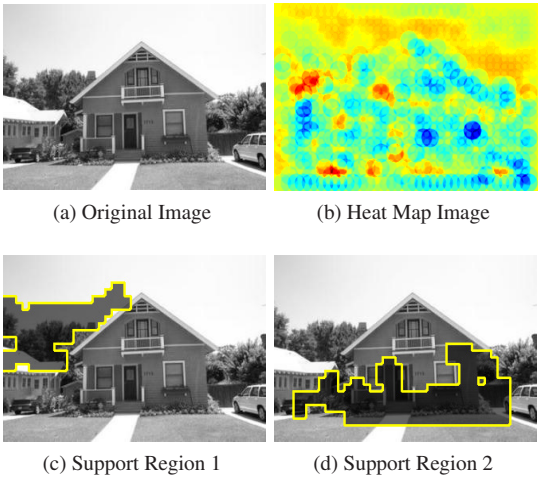
Lei Wang
School of Computer Science & Software Engineering
University of Wollongong, NSW 2522, Australia
leiw@uow.edu.au

## Abstract

*In the past decade, the bag-of-feature model has established itself as the state-of-the-art method in various visual classification tasks. Despite its simplicity and high performance, it normally works as a black box and the classification rule is not transparent to users. However, to better understand the classification process, it is favorable to look into the black box to see how an image is recognized. To fill this gap, we developed a tool called Restricted Support Region Set (RSRS) Detection which can be utilized to visualize the image regions that are critical to the classification decision. More specifically, we define the Restricted Support Region Set for a given image as such a set of size-restricted and non-overlapped regions that if any one of them is removed the image will be wrongly classified. Focusing on the state-of-the-art bag-of-feature classification system, we developed an efficient RSRS detection algorithm and discussed its applications. We showed that it can be used to identify the limitation of a classifier, predict its failure mode, discover the classification rules and reveal the database bias. Moreover, as experimentally demonstrated, this tool also enables common users to efficiently tune the classifier by removing the inappropriate support regions, which can lead to a better generalization performance.*

## 1. Introduction

Classification based on bag-of-feature (BoF) model [11, 2] has become very popular in the past several years. It works surprisingly well in various classification tasks, e.g. object [2], scene [8], action [7], human pose [10], subjective properties [4], to name a few. By simply following local feature extraction, coding, and pooling three steps, an image is represented as a fixed length vector and a classifier can be learned [1, 15, 14, 9]. Despite the simplicity and high performance, the BoF model usually works as a black box and the classification rule is not transparent to users. However, it is favorable to look into the black box and visualize *how an image is classified* in many occasions. For example, for many computer vision applications in which human is involved, e.g. computer-aided medical image diagnosis, it is highly desirable that the classification system can provide information about how a decision is made rather than simply giving a positive/negative result. More generally, this visualization can also facilitate users to identify the potential problems and improve the classification system.



(a) Original Image (b) Heat Map Image

(c) Support Region 1 (d) Support Region 2

Figure 1: The comparison between our Support Region Visualization and the existing Heat Map [19] visualization. (a). Original Image; (b). Heat Map Image; (c),(d) **Two support regions detected for this image**

Based on the above motivations, we proposed a visualization method by examining the dependency between the presence/absence of a particular image region and the classification decision. Formally, we put forward a concept named "Restricted Support Region Set" (RSRS) for a given image. It is defined as a set of size-restricted and non-overlapped regions having the property that *if any one of*

*the regions is removed, this image will be wrongly classified.* The idea behind is that if the absence of a particular region makes the classification result flip from right to wrong, this region must have provided a supporting role for correctly recognizing the image. Thus by examining whether these regions are really related to the visual concept to be learned, we can tell whether the classifier has learned the visual concept as required.

As will be shown in Section 3.2, the restricted support region set is system-specific. In this paper, we mainly focus on developing a RSRS detection algorithm for the state-of-the-art classification systems which use dense sampled local features, with sparse coding coefficients, max-pooling and linear classifier. Nevertheless, the concept of the restricted support region set and its detection principle can be applied to general classification systems.

We also show examples about the applications of this tools: by examining the support regions of a single image, we can predict under what kind of occlusion the image will be misclassified. By conducting experiments on PASCAL VOC 2007 dataset, we reveal some interesting phenomenon about the classification rules based on the support region locations in multiple images. We also employ this tool as an interactive interface which enables users to remove the inappropriate support regions to generate new training samples. Adding these new samples to the training set and retraining, the generalization performance of the classifier can be improved as discovered in our experiment.

## 2. Related Work

In the literature, there are mainly two ways to visualize the classification rule. The first one is to visualize the prototypes which are useful for classification. Examples include visualizing the learned part in shape model [3], mined prototypes [18], or to be more relevant to the BoF model, showing the patches assigned to the visual words with maximal inter-category discrimination [5]. While this kind of visualization attempts to directly display the visual components whose occurrences will increase the confidence of predicting the presence of the object, it will become less practical when the codebook size becomes larger, for example, several thousands, or when the feature dimension is amplified by using Spatial Pyramids [8]. The other way is to display the highly weighted interest points on the image [17] or use the heat map [19] when dense sampled local feature is used. An example of heat map is shown in Figure 1 where the warmer color indicates the higher weight. The drawbacks of this visualization method are two folds: (1) although it shows the highly weighted region, it is still unclear how important these regions are, e.g. whether they are so crucial that without them the image will not be correctly classified? (2) In max-pooling where multiple occurrences of a same visual word are only counted once (or only the maximum coding coefficient is recorded), the importance of a local patch is determined by two factors: (a) the word (or coding) to which the patch is quantized into; (b) whether there is another patch quantized into the same word and its coding coefficient is larger. The heat map, which only considers the first factor, becomes less accurate in this scenario. For example, a homogeneous region such as the sky region in Figure 1 may have many patches assigned to the same word (coding), hence removing a large portion of it will have no impact on the classification decision score as long as there is still one patch assigned to the same visual word left (or the patch with the largest coding coefficient is not removed). However, all the patches in the sky region are shown with equal importance in heat map.

The detection of restricted support region set in this paper also shares similar spirit with the problem of efficiently searching for the most discriminative window/sub-region [6]. At the first glance their methods can be readily applied to the classification rule visualization. However, their efficient solutions assume the linearity, that is, the sum of classification decision scores of two separate regions equals to the score of the region obtained by merging them together. However, this assumption will not hold anymore in the state-of-the-art classification systems where max-pooling is used. This is because in max-pooling the "discriminativeness" of one region, which can be evaluated by the change of classification decision score after removing it, is not solely determined by the region itself but also by the remaining parts of the image (for the reason, see the factor (b) above). So we cannot evaluate the "discriminativeness" of one region by simply summing the contribution of each sub-region within but have to calculate it dynamically as shown by the algorithm developed in this work.

Finally, our work is also related to a recent study on annotator's rationales [4]. However, its purpose is to incorporate the annotator's rationale to design a better classifier, while our aim is to visualize the "rationales of a classifier" learned purely from a set of training samples.

## 3. Support Region Detection

### 3.1. Background and Notations

Let $\mathcal{I} = \{\mathcal{I}_1, \mathcal{I}_2, \cdots, \mathcal{I}_M\}$ denote a set of image samples. In the BoF model, a set of local features $\{\mathbf{x}_1^i, \mathbf{x}_2^i, \cdots, \mathbf{x}_{N_i}^i\}$ are extracted from each image $\mathcal{I}_i$ and they are encoded with a learned dictionary $\mathbf{B} \in \mathbb{R}^{d \times V}$. This step is called "coding", which maps the local feature $\mathbf{x}_j^i \in \mathbb{R}^d$ to the coding coefficient $\mathbf{u}_j^i \in \mathbb{R}^V$. In this paper we assume $\mathbf{u}_j^i \geq 0$ and this is true for most existing coding methods [15, 14, 9]. Recent studies [15, 14, 9] show that assigning a local feature to a small number of visual words to produce sparse coding coefficients can boost recognition performance. To obtain image-level representa-

tion, the coding coefficients of all local features in an image are pooled together. Two pooling strategies are usually used: sum-pooling and max-pooling. The former sums all the coding coefficients to obtain the image representation $\mathbf{z}^i = \sum_{j=1}^{N_i} \mathbf{u}_j^i$ while the latter computes the dimension-wise maximum, that is, $z_k^i = \max_j u_{jk}^i$, where $z_k^i$ and $u_{jk}^i$ denote the $k$th component of $\mathbf{z}^i$ and the $k$th component of $\mathbf{u}_j^i$, respectively. It has been shown [1] that max-pooling can produce the state-of-the-art performance by using a simple linear SVM classifier and it significantly outperforms sum-pooling in this situation. Hence, in this paper we focus on the following setting: coding methods producing sparse coding coefficient [15, 14, 9], max-pooling, and linear classifier.

### 3.2. Basic Definitions

Let $\mathcal{F} : \mathcal{I} \to \mathbb{R}^V$ denote the mapping from an image to its image-level representation. A classifier can then be expressed as a mapping $\mathcal{C} : \mathbb{R}^V \to \mathbb{Z}$ from the representation to the predicted class label $\hat{y}$. Combining the two mappings, we have $\hat{y} = \mathcal{C}(\mathcal{F}(\mathcal{I}))$. Let's focus on binary classification since multi-class classification can be decomposed into multiple one-vs-rest binary classifications. Let $y_i \in \{-1, 1\}$ be the ground truth class label of the $i$th image, where $y_i = 1$ denotes the positive sample in which the object is present. Viewing an image as an array of pixels, we define *connected region* as a set of pixels such that there exists a inner path connecting any pair of pixels in this set. One can use the 4- or 8-neighborhood to define the connectivity. Then a support region can be formally defined as follows:

**Definition 1** *For any correctly classified image $\mathcal{I}_i \in \{\mathcal{I}_l | \mathcal{C}(\mathcal{F}(\mathcal{I}_l)) = y_l\}$, if there exists a connected region $\mathcal{R}_s$ such that removing it only will make the image wrongly classified, that is, $\mathcal{C}(\mathcal{F}(\mathcal{I}_l - \mathcal{R}_s)) \neq y_l$, this region will be called a support region of image $\mathcal{I}_i$ w.r.t the given image representation $\mathcal{F}$ and classifier $\mathcal{C}$, denoted as $\mathcal{R}_s$ of $(\mathcal{I}_i, \mathcal{F}, \mathcal{C})$.*

Here, the minus operator in $\mathcal{I}_l - \mathcal{R}_s$ denotes the set difference. Note that we do not define a support region for incorrectly classified images because for those images the misclassification may be due to the lack of certain regions. We are not able to visualize the missing regions since we cannot create it from nowhere. Also, we focus on detecting the support regions for the correctly classified "positive" samples, that is, $y_i = \hat{y}_i = 1$, because for those samples support regions are often more meaningful. They are expected to be the key parts of the object/scene and therefore are easier for human to understand and evaluate the behavior of classifier.

Note that if $\mathcal{R}_s$ is a support region, its superset could be a support region too. As a result, many redundant support regions could be generated by enlarging a current support region or shifting it slightly. To avoid this situation, we further define *Restricted Support Region Set* by introducing two more constraints:

**Definition 2** *The Restricted Support Region Set (RSRS) $\mathcal{R}_{rs}$ of $(\mathcal{I}_i, \mathcal{F}, \mathcal{C})$ is defined as a set of regions that satisfy the following three conditions: (1) any region in the set is a support region of $(\mathcal{I}_i, \mathcal{F}, \mathcal{C})$; (2) any pair of regions in the set are not spatially overlapped; (3) the size of each region is less than a predefined threshold.*

The region size constraint also improves the detection efficiency and avoids generating a meaningless over-large region, e.g. the whole image as a support region. The maximum region size is set as large as that we believe human will not be able to tell the content of the image after removing such a large region.

### 3.3. Detecting Restricted Support Region Set

As seen, the definition of RSRS depends on the image representation $\mathcal{F}$ and image classifier $\mathcal{C}$. In this paper, our focus is to develop RSRS detection algorithm for the state-of-the-art image classification system introduced in 3.1. Recall that $\mathbf{u}_j^i$ is the coding coefficient for the $j$th local feature in the $i$th image (For the notation simplicity, we omit the superscript $i$ in $\mathbf{u}_j^i$ from now on). Let $u_{jk}$ be the $k$-th dimension of the coding coefficient $\mathbf{u}_j$ and $\mathbf{w}, b$ be the linear SVM classifier parameters. The decision function is:

$$\hat{y} = \text{sgn}\left(\sum_{k=1}^{V} w_k \max_j \{u_{jk}\} + b\right) \tag{1}$$

Note that the max-operator is due to the use of max-pooling. To compute the decision value after removing a region, we simply do not take the coding coefficients extracted from the region into account. Focusing on the case $\hat{y} = 1$, a support region $\mathcal{R}_s$ is mathematically expressed as:

$$\sum_{k=1}^{V} w_k \max_{\{j | \mathcal{P}_j \notin \mathcal{R}_s\}} \{u_{jk}\} + b < 0; \;\; where \; |\mathcal{R}_s| \leq \mathrm{A}_0 \tag{2}$$

which indicates that $\hat{y}$ in (1) changes from +1 to -1 after removing the region $\mathcal{R}_s$ whose size is thresholded by $\mathrm{A}_0$. Here, $\mathcal{P}_j$ denotes the spatial location of the $j$th local feature.

To detect the restricted support region set, we employ a sequential detection approach, that is, we first detect one support region and then detect another one outside this region. This procedure is repeated until no valid support region can be found. More specifically, a region growing algorithm is used to form a valid support region from a given seed. Thanks to the sparseness of coding coefficient, the decision value can be updated very efficiently without recalculating (1) after a new pixel is added to the current support region. To explain this algorithm, let's first define

$$\mathcal{J}(\mathcal{R}_p, \mathcal{R}_q) = \sum_{k=1}^{V} w_k \left( \max_{\{j|\mathcal{P}_j \in \mathcal{R}_p\}} \{u_{jk}\} - \max_{\{j|\mathcal{P}_j \in \mathcal{R}_p - \mathcal{R}_q\}} \{u_{jk}\} \right) \tag{3}$$

where $\mathcal{R}_p$ and $\mathcal{R}_q$ are two image regions satisfying $\mathcal{R}_q \subset \mathcal{R}_p$. By letting $\mathrm{S}_0 = \sum_{k=1}^{V} w_k \max_j \{u_{jk}\} + b$, we can redefine (2) as:

$$\mathcal{J}(\mathcal{I}, \mathcal{R}_s) \geq \mathrm{S}_0; \;\; where \;\; |\mathcal{R}_s| \leq \mathrm{A}_0 \tag{4}$$

In the process of region growing, the support region grows in an iterative way: $\mathcal{R}_t = \mathcal{R}_{t-1} \cup \mathcal{P}_t$, where $\mathcal{P}_t$ denotes a new pixel [1] added to the current region in each iteration. Noting that $\mathcal{R}_t = \mathcal{P}_1 \cup \mathcal{P}_2 \cdots \cup \mathcal{P}_t$, $\mathcal{J}(\mathcal{I}, \mathcal{R}_t)$ (let $\mathcal{R}_0 = \emptyset$) can be recursively computed as:

$$\mathcal{J}(\mathcal{I}, \mathcal{R}_t) = \mathcal{J}(\mathcal{I}, \mathcal{R}_{t-1}) + \mathcal{J}(\mathcal{I} - \mathcal{R}_{t-1}, \mathcal{P}_t). \tag{5}$$

$\mathcal{P}_t$ is selected from the boundary points of current support region based on the following criterion:

$$\hat{\mathcal{P}}_t = \underset{\mathcal{P}_t \in \mathrm{Boundary}\{\mathcal{R}_{t-1}\}}{\operatorname{argmax}} \mathcal{J}(\mathcal{I} - \mathcal{R}_{t-1}, \mathcal{P}_t). \tag{6}$$

Note that $\mathcal{J}(\mathcal{I} - \mathcal{R}_{t-1}, \mathcal{P}_t)$ can be written in the form of (3). However, we do not need to calculate the summation, $\sum_{k=1}^{V}$, for every dimension $k$ but only evaluate those corresponding to the very few nonzero coding coefficients of $\mathcal{P}_t$. This is because for all dimensions where $\mathcal{P}_t$ has zero-valued coding coefficient, the term $\left(\max_{\{j|\mathcal{P}_j \in \mathcal{I} - \mathcal{R}_{t-1}\}} \{u_{jk}\} - \max_{\{j|\mathcal{P}_j \in \mathcal{I} - \mathcal{R}_{t-1} - \mathcal{P}_t\}} \{u_{jk}\}\right)$ will be zero and thus can be omitted. Hence, we have:

$$\begin{aligned} &\mathcal{J}(\mathcal{I} - \mathcal{R}_{t-1}, \mathcal{P}_t) \\ &= \sum_{k|u_{tk} \neq 0} w_k \left( \max_{\{j|\mathcal{P}_j \in \mathcal{I} - \mathcal{R}_{t-1}\}} \{u_{jk}\} - \max_{\{j|\mathcal{P}_j \in \mathcal{I} - \mathcal{R}_{t-1} - \mathcal{P}_t\}} \{u_{jk}\} \right) \end{aligned} \tag{7}$$

Moreover, note that only when the maximum of $\{u_{jk}\}$ over $\{j|\mathcal{P}_j \in (\mathcal{I} - \mathcal{R}_{t-1})\}$ is produced by the newly added pixel $\mathcal{P}_t = \mathcal{R}_t - \mathcal{R}_{t-1}$, can the difference term $\max_{\{j|\mathcal{P}_j \in \mathcal{I} - \mathcal{R}_{t-1}\}} \{u_{jk}\} - \max_{\{j|\mathcal{P}_j \in \mathcal{I} - \mathcal{R}_t\}} \{u_{jk}\}$ become nonzero. In such a situation, the maximum of $\{u_{jk}\}$ over $\{j|\mathcal{P}_j \in (\mathcal{I} - \mathcal{R}_t)\}$ will equal to the second maximum (we regard the duplicated maximum as second maximum too) in the region $(\mathcal{I} - \mathcal{R}_{t-1})$.

Hence, to efficiently compute (7), two tables storing $\max_{\{j|\mathcal{P}_j \in \mathcal{I} - \mathcal{R}_{t-1}\}} \{u_{jk}\}$ and $\max 2_{\{j|\mathcal{P}_j \in \mathcal{I} - \mathcal{R}_{t-1}\}} \{u_{jk}\}$ are maintained, where $\max 2$ denotes the second maximum. Once a pixel $\mathcal{P}_t$ is added to the existing region $\mathcal{R}_{t-1}$, we can then quickly check whether its associated $u_{tk}$ $(k = 1, 2, \cdots, V)$ is the maximum over the region $(\mathcal{I} - \mathcal{R}_{t-1})$. If it is, then the above difference term can be calculated as $\max_{\{j|\mathcal{P}_j \in \mathcal{I} - \mathcal{R}_{t-1}\}} \{u_{jk}\} - \max 2_{\{j|\mathcal{P}_j \in \mathcal{I} - \mathcal{R}_{t-1}\}} \{u_{jk}\}$, and zero otherwise. These two tables are updated once a new optimal growing point $\hat{\mathcal{P}}_t$ is identified and added. Again, since adding one pixel only involves the change in few dimensions, the update will be very efficient. The overall region growing algorithm is shown in Algorithm 1.

**Algorithm 1** Grow A Single Support Region

**Require:** A seed point $\mathcal{P}_1$ and $\mathrm{S}_0$, $\mathrm{A}_0$

1: $\mathcal{R}_1 \leftarrow \{\mathcal{P}_1\}$ , $\mathrm{S} \leftarrow 0$ , $\mathrm{A} \leftarrow 0$
2: Build two tables
$\mathbf{T}_1(\mathcal{R}_1) = \max_{\{j|\mathcal{P}_j \in \mathcal{I} - \mathcal{R}_1\}} \{u_{jk}\}$
$\mathbf{T}_2(\mathcal{R}_1) = j|\max 2_{\{\mathcal{P}_j \in \mathcal{I} - \mathcal{R}_1\}} \{u_{jk}\}$ ;
3: **while** $\mathrm{S} \leq \mathrm{S}_0$ and $\mathrm{A} \leq \mathrm{A}_0$ **do**
4: $\hat{\mathcal{P}}_t \leftarrow \underset{\mathcal{P}_t \in \mathrm{Boundary}\{\mathcal{R}_{t-1}\}}{\operatorname{argmax}} \mathcal{J}(\mathcal{I} - \mathcal{R}_{t-1}, \mathcal{P}_t)$
5: $\mathbf{T}_i(\mathcal{R}_t) \leftarrow \mathbf{T}_i(\mathcal{R}_{t-1} \cup \hat{\mathcal{P}}_t) \quad i = 1, 2$
6: $\mathrm{A} \leftarrow \mathrm{A} + 1$ and $\mathrm{S} \leftarrow \mathrm{S} + \mathcal{J}(\mathcal{I} - \mathcal{R}_{t-1}, \hat{\mathcal{P}}_t)$ according to (7).
7: **end while**
8: Return success if (2) is satisfied, that is, $\mathrm{S} \geq \mathrm{S}_0$ and failure otherwise.

The seed point $\mathcal{P}_1$ in Algorithm 1 can be selected in various ways including random selection. However, not every seed point can successfully grow a support region under the region size constraint. To reduce the chance of the support region growing failure, we develop a heuristic method to seek the most promising seed points. The method first tests $\mathcal{J}(\mathcal{I}, \mathcal{P}_0)$ for every pixel $\mathcal{P}_0$ and selects the $K$ points[2] with top $K$ largest $\mathcal{J}$ values as the initial seeds. Once a valid support region is detected, the seed points falling into the detected region will be removed and new $K$ seeds are generated outside the region. The region growing is stopped when no valid support region can be detected in the remaining part of the image or the number of trials has reached its limitation. We can use the upper bound in (8) to quickly assess the possibility of finding a valid support region in the remaining image region $\mathcal{I}'$.

$$\mathcal{J}(\mathcal{I}, \mathcal{R}_s) \leq \sum_{k|w_k \geq 0} w_k \left( \max_{i|\mathcal{P}_i \in \mathcal{I}'} \{u_{ik}\} \right); \;\; where \;\; \mathcal{R}_s \subset \mathcal{I}' \tag{8}$$

Obviously, if we find the upper bound (RHS of (8)) has been less than $\mathrm{S}_0$ in (4), then it will not be possible to find a support region $\mathcal{R}_s$ in $\mathcal{I}'$. Combining all these steps, the RSRS detection algorithm is summarized in Algorithm 2.

[1] Since local patches are densely sampled, a pixel referred here actually means a patch center. It corresponds to a small block in the original image.

[2] In practice, we require these points not be too close to each other.

**Algorithm 2** Detect Restricted Support Region Set (Multiple Support Regions)

**while** RHS (8) is no less than $S_0$ **do**
  Select top $K$ points with top $K$ largest $\mathcal{J}$ values from the remaining part of image as the seed points.
  **for** $j = 1$ to $K$ **do**
    Grow one region with seed point $\mathcal{P}_j$ by using Algorithm 1.
    **if** success **then**
      Report one support region;
      break;
    **end if**
  **end for**
**end while**

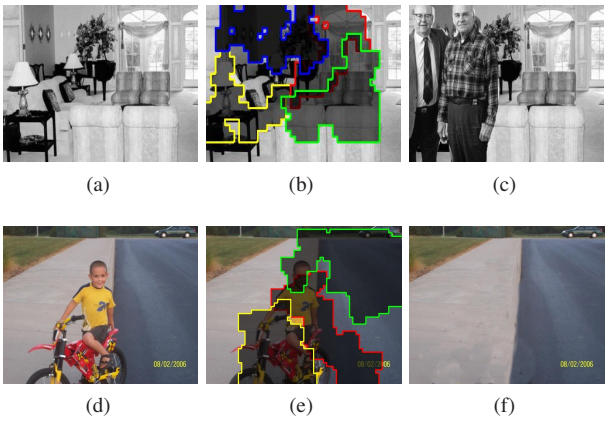


Figure 2: (a)(d) Original Images: "living room" in Scene-15 and "car" in PASCAL respectively. (b)(e) multiple detected support regions. (c)(f) synthesized images with one support region replaced by new content.

## 4. Applications and Examples

In this section, three applications of the proposed RSRS detection are discussed with examples. They are: (1) predict the failure mode of classifier. We give two examples from the commonly used scene and object classification datasets: Scene-15 and PASCAL VOC 2007 respectively. (2) understand the classification rule and discover the database bias [12]. We demonstrate this application on PASCAL dataset because it has the ground truth bounding boxes which enable us to make quantitative analysis. (3) interactively deselect inappropriate support regions and generate new training samples. We use Graz02 and a subset of PASCAL containing the same three classes *bike, car, person* in Graz02. More details about this experiment protocol are discussed in Section 4.3.

For the first and second applications, we extract the HOG local feature and apply the Spatial Pyramid by following the setting in [8] and [16] respectively. For the third application, we also extract HOG for the Graz02 with the setting of [8] but without using the Spatial Pyramid because there is significant translation of object locations in each image. The same settings are also used for the three-class subset of PASCAL dataset to facilitate the cross-dataset performance test in this application. In all examples, we use linear SVM as the classifier and use the localized soft-assignment coding [9] due to its simplicity and high performance.

The maximum region size $A_0$ is set to 200 (patches) for Scene 15 and 300 (patches) for the other two datasets due to the different image sizes. Please note that the detected support region size can be much smaller than $A_0$ and the number of detected support regions varies automatically with image content.

### 4.1. Application I: Predict the Failure Mode of Classifier

A straightforward application of RSRS detection is to predict the failure mode of classifier. According to the definition, if a support region is removed the image will be wrongly classified. Hence, we could expect that misclassification will happen if the support region is replaced with an object irrelevant to the to-be-recognized visual concept. Figure 2 gives such an example with two images from the training set in Scene-15 category "*living room*" and PASCAL category "*car*" respectively. As shown in Figure 2 (b)(e), the detected support regions [3] suggest that the left corner region with a lamp is indispensable for the correct classification of the living room image and the correct classification of the car image heavily relies on the region with a bike and a kid. We may predict that if these two regions are replaced, the images will be misclassified. To verify this prediction, we produce two "real world alike" synthesized images with the support regions replaced as shown in the last row of Figure 2. After re-running the classification system on these synthesized images, the classification decision scores drop from 0.9988 to -0.012 and from 1.0276 to -0.3206 for the *living room* and *car* images respectively. In both cases, the images cannot be correctly classified anymore. However, as can be seen from the last row of Figure 2, we, as human, can still correctly classify these two images.

[3] Since the "pixel" in Algorithm 1 is actually the patch center, the non-overlapping constraint of RSRS in effect requires no overlapping among patch centers. That is why small overlapping is observed in multiple support regions shown in Figure 2 (b)(e).

(a) chair (b) aeroplane (c) cow (d) sheep (e) bus (f) TV/monitor (g) train (h) boat

Figure 3: Some typical support regions detected on PASCAL dataset, see text for more detail. For the reference convenience, only one support region is displayed in each image even when there are more support regions in the same image.

### 4.2. Application II: Understand the Classification and Discover Database bias

To better understand the classification system, we can investigate the support region locations in multiple images and see *what kinds of objects are frequently covered by the support regions*. Then we can have some intuitive idea about what visual cues are important for predicting the presence of the object to be recognized. To demonstrate this application, we use PASCAL dataset as an example. It contains 20 object bounding box annotation for each image and this allows us to perform some quantitative analysis. We first present some examples of detected support regions in Figure 3, including their ground truth class labels. By checking the locations of the support regions over the whole training set, we summarize them as three typical cases: (1) when the classification decision score is close to zero, that is, the sample is close to the decision hyperplane, many support regions will be detected because it is easy to find a small region removing which makes classification incorrect. Many of the detected support regions in this case are not relevant to the visual concept to be learned. Figure 3 (a) and (b) show one of such regions in each image. (2) The support region exactly covers the object to be recognized, such as in Figure 3 (c) and (d). It normally happens when the object is clearly visible or the background is simple. (3) The support region covers an object/scene which is correlated (co-exists) with the to-be-recognized object in the training set. Examples can be seen in Figure 3 (e) to (h). These objects/scenes may be interpreted as the "context" [13]. Some context is reasonable such as the harbor and water for boat recognition as in (h). But some are less reasonable like person for TV/monitor recognition. The dependency in the latter case can be due to dataset bias. The images of PASCAL is collected from the Internet and people are normally less interested in uploading an image solely about a TV/monitor. Therefore, for many images on Internet the TV/monitor may happen to occur in the scenario like "people in the living room" and high correlation between TV/monitor and person is then resulted.

To quantitatively evaluate the role of context and object in the image classification, we compute two statistics with

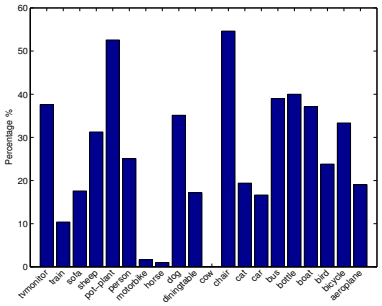



Figure 4: The percentage of images with at least one support region not covering the object to be recognized.

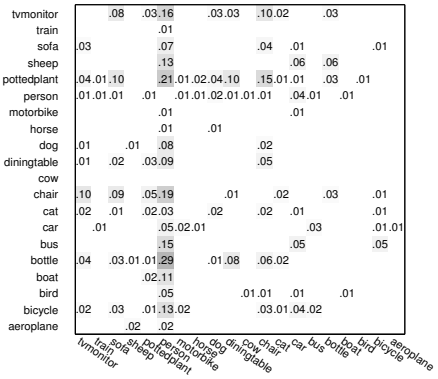



Figure 5: The percentage of images with at least one support region covering the 20 objects, showing in each row for a category. The blank entry indicates value 0.

the help of the bounding box in the PASCAL dataset. The first one counts the percentage of images with at least one support region that does not have any overlap with the object bounding box. This measurement indicates the dependency of correct classification on the context, because these support regions cover the "context" and without their help the correct classification cannot be made. This is true even if all the remaining support regions all cover the object. As shown in Figure 4, the percentage suggests that for most categories, the classifier heavily (more than 20% of samples) relies on the context visual cues. For categories like *chairs* and *pot-plants*, the figure suggests that more than half of the images in the category will be misclassified if a context region is removed. Only three categories *motorbike, horse,*

*cow* have no or little dependency on the context. Examining the corresponding images, we found they usually have a clear spatial layout and homogeneous background.

We then further ask *what kinds of objects are covered by the support regions that have no overlapping with the object to be recognized*. To answer this question, we need to access the bounding box annotation of all objects in all images. Since we only have the bounding boxes for 20 objects, we restrict our quantitative analysis on these objects. For each category, we calculate the percentage of images with at least one support region covering[4] the objects other than the to-be-recognized object, as shown in the each row of Figure 5. Some interesting correlation can be seen: (1) the *person* category takes a high percentage for many categories, which means that the presence of person provides a strong visual cue in correctly classifying these categories. Taking *TV/monitor* class for example, it suggests that 16% of the training images will be misclassified if the person is removed. For *pot-plant, chair and bottle*, the percentage w.r.t *person* is the highest. Checking their images, we do find that these objects frequently co-occur with person.

Generally, context may benefit recognition, especially when an object is small and has no discriminative appearance. However, relying too much on the context may also bring risks since the context may come from an unreasonable correlation introduced by a biased dataset. It is also noteworthy that the dependency between objects and their context valid in one domain may become invalid in another domain. For example, the presence of highway may provide strong visual clue for detecting a car. But this context could be harmful for the application such as recognizing whether there is a car on the road or not.

### 4.3. Application III: Interactive Support Region Selection

The third application of RSRS detection is to use it as an interface for interactively de-selecting inappropriate support regions. The basic idea is that if a user does not think a support region reasonable, that is, without this region the image should still be recognizable, then the user can generate a new training sample by removing this region, assign its class label to +1, and add it to the original training set to retrain the classifier. To test this idea, we carry on an experiment on Graz02 and the three-class subset of PASCAL dataset. Our experiment protocol is to train a classifier on Graz02 and test it on both Graz02 and the PASCAL subset. The reason of this protocol is: Both datasets have the three categories of *person, car* and *bike*, but the context of the three objects in the two datasets are quite different, as shown in the first and second rows of Figure 6. Comparing with PASCAL, the images in Graz02 are mostly taken in an outdoor environment. Therefore, the context which is useful for Graz02 may not be useful in classifying images in PASCAL anymore. So if we can force the classifier to focus more on the to-be-recognized object rather than the context, we may expect a performance improvement on the cross-dataset test. To validate this, the classifier is built in two schemes: (1) directly train it on the training set of Graz02. (2) Train an initial classifier on the Graz02 training set and display the detected support regions. Then users are allowed to de-select the regions not covering the to-be-recognized object and new samples are generated. Then we retrain the classifier by adding these new samples to the Graz02 training set and the performance of these two schemes is compared. In detail, the Graz02 dataset contains four class of *car, bike, person* and *background* four classes. We use *background* as the negative class and *car, bike, person* as the positive class, respectively, to form three binary classification tasks. For each task, half of images in the two involved classes are used for training and the remaining is for test. For PASCAL, we extract the *car, person, bike* images from its training/validation set to form the positive classes respectively, and for each of them, we extract the same number of images from the remaining classes to from a negative class.

Table 1: Experimental result on Graz-Pascal dataset. Evaluated by average precision.

| Test Setting | Bike | Car | Person |
|---|---|---|---|
| Without new samples Test on Graz | 93.3 | 79.2 | 86.4 |
| **With new samples Test on Graz** | 93.8 | 79.7 | **88.3** |
| Without new samples Test on Pascal | 73.2 | 77.5 | 67.7 |
| **With new samples Test on Pascal** | 73.4 | **80.9** | **68.4** |

Examples of deselected support regions are shown in the last row of Figure 6. If multiple support regions are deselected in an image, we generate a single sample by removing all of them. After scanning all the images from the training set of Graz02, we generate 40, 60, 150 new samples for *bike, person* and *car* respectively. The result is shown in Table 1. As seen, adding the new samples does improve the classification performance on several occasions: *car* in PASCAL, *person* in both PASCAL and Graz02. The largest improvement is observed on *car* in PASCAL. It is quite reasonable because we observed much more images with inappropriate support regions from *car* in Graz02, which implies the classifier learned on Graz02 may rely more heavily on the context. Consequently, a better generalization perfor-

[4]By covering, we mean a support region occupies more than 20% area of the object bounding box.

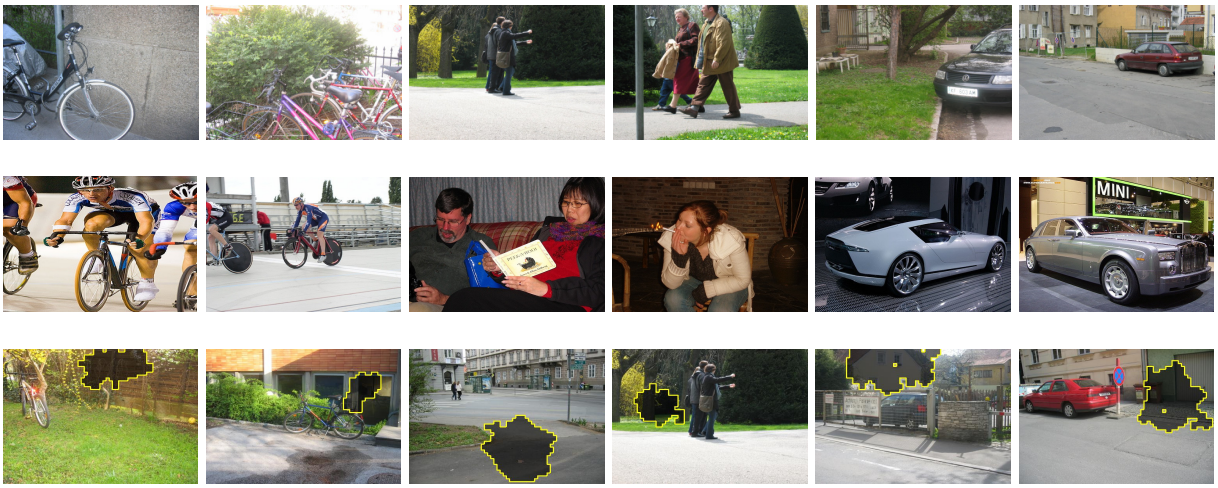



Figure 6: First row: example images from Graz02. Second row: example images from PASCAL three class (*car, bike, person*) subset. Last row: example inappropriate support regions in Graz02 that are deselected to generate new samples.

mance can be obtained by "reducing" this dependence via retraining with the new samples. This also explains why no significant improvement is observed on *bike* for which the number of newly generated samples is the smallest. It is also interesting to note that adding new samples does not lead to much improvement on the test set of *car* in Graz02. This may suggest that in this case the context visual cues learned from the training set is also useful to the test set.

## 5. Conclusion

This paper developed a tool, called Restricted Support Region Set Detection and Visualization, to understand the Bag-of-feature image classification system. This tool has been used to predict the failure mode of classifier, understand the classification, reveal dataset bias, and interactively generate new samples to improve generalization performance. Through the interesting phenomenon discovered by our tool, this study provides more insights on how different visual cues contribute to the classification of images.

## 6. Acknowledgement

Lei Wang is partly supported by the ARC grant LP0991757 and Lingqiao Liu thanks this grant for supporting part of his travel cost.

## References

[1] Y.-L. Boureau, F. Bach, Y. LeCun, and J. Ponce. Learning mid-level features for recognition. *CVPR 2010*. 1, 3

[2] G. Csurka, C. R. Dance, L. Fan, J. Willamowski, and C. Bray. Visual categorization with bags of keypoints. In *ECCV 2004*. 1

[3] R. Fergus, P. Perona, and A. Zisserman. Object class recognition by unsupervised scale-invariant learning. In *CVPR 2003*. 2

[4] K. G. Jeff Donahue. Annotator rationales for visual recognition. In *ICCV*, 2011. 1, 2

[5] F. Jurie and B. Triggs. Creating efficient codebooks for visual recognition. In *ICCV 2005*. 2

[6] C. H. Lampert, M. B. Blaschko, and T. Hofmann. Beyond sliding windows: Object localization by efficient subwindow search. In *CVPR 2008*. 2

[7] I. Laptev, M. Marszalek, C. Schmid, and B. Rozenfeld. Learning realistic human actions from movies. *CVPR*, 2008. 1

[8] S. Lazebnik, C. Schmid, and J. Ponce. Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. *CVPR*, 2, 2006. 1, 2, 5

[9] L. Liu, L. Wang, and X. Liu. In defence of soft-assignment coding. *ICCV 2011*. 1, 2, 3, 5

[10] H. Ning, W. Xu, Y. Gong, and T. Huang. Discriminative learning of visual words for 3d human pose estimation. In *CVPR*, 2008. 1

[11] J. Sivic and A. Zisserman. Video Google: A text retrieval approach to object matching in videos. In *ICCV 2003*. 1

[12] A. Torralba and A. A. Efros. Unbiased look at dataset bias. In *CVPR 2011*. 5

[13] J. R. R. Uijlings and A. W. M. Smeulders1. What is the spatial extent of an object? In *CVPR 2009*. 6

[14] J. Wang, J. Yang, K. Yu, F. Lv, T. Huang, and Y. Gong. Locality-constrained linear coding for image classification. *CVPR 2010*. 1, 2, 3

[15] J. Yang, K. Yu, Y. Gong, and T. S. Huang. Linear spatial pyramid matching using sparse coding for image classification. In *CVPR*, 2009. 1, 2, 3

[16] J. Yang, K. Yu, and T. Huang. Efficient highly over-complete sparse coding using a mixture model. In *ECCV 2010*. 5

[17] L. Yang, R. Jin, R. Sukthankar, and F. Jurie. Unifying discriminative visual codebook generation with classifier training for object category recognition. In *CVPR*, 2008. 2

[18] B. Yao and L. Fei-Fei. Grouplet: A structured image representation for recognizing human and object interactions. In *CVPR*, San Francisco, USA, 2010. 2

[19] B. Yao, A. Khosla, and L. Fei-Fei. Combining randomization and discrimination for fine-grained image categorization. In *CVPR*, 2011. 1, 2