A Classifier-guided Approach for Top-down Salient Object Detection

Hisham Cholakkal, Jubin Johnson, Deepu Rajan

School of Computer Engineering,
Nanyang Technological University, Singapore

Abstract

We propose a framework for top-down salient object detection that incorporates a tightly coupled image classification module. The classifier is trained on novel category-aware sparse codes computed on object dictionaries used for saliency modeling. A misclassification indicates that the corresponding saliency model is inaccurate. Hence, the classifier selects images for which the saliency models need to be updated. The category-aware sparse coding produces better image classification accuracy as compared to conventional sparse coding with a reduced computational complexity. A saliency-weighted max-pooling is proposed to improve image classification, which is further used to refine the saliency maps. Experimental results on Graz-02 and PASCAL VOC-07 datasets demonstrate the effectiveness of salient object detection. Although the role of the classifier is to support salient object detection, we evaluate its performance in image classification and also illustrate the utility of thresholded saliency maps for image segmentation.

Keywords: Saliency, Top-down saliency, Image classification, Semantic segmentation

1. Introduction

Visual attention in humans is largely affected by past experiences of the visual world [1]. This phenomenon, includes *contextual cuing* which helps human brain to respond faster by spending less neural resources at spatial locations where the probability of a target is low. Mimicking contextual cuing, top-down saliency approaches learn the relevance of image features and spatial locations from a set of training images which helps to reduce the search space for computationally expensive tasks such as object segmentation and detection. On the other hand, in bottom-up saliency detection, visual rarity is used to identify salient regions and hence, it often fails in the presence of background clutter due to which the salient object does not 'pop-out'.

In general, there are two types of top-down saliency detection approaches– knowledge-based and task-based [1]. Knowledge-based approaches aim to learn the prior knowledge about relevant objects under free-viewing conditions [2, 3, 4, 5, 6]. e.g., attention to faces. Task-based top-down saliency models use prior knowledge to infer a probability map that suppresses image regions not relevant to the task [7, 8, 9, 10]. Our approach falls under the second category, where the task is estimating the presence of objects from a particular category in an image (image classification) as well as identifying the image locations of those objects (object localization). The regions that are closer to the pre-learnt priors are assigned higher saliency values, even if they are not salient in the 'pop-out' sense.

Salient object detection is different from object detection in that the latter aims to produce a tight rectangular bounding box around the object of interest. Even for a moderately sized image, the algorithm needs to sample and classify millions of rectangular boxes with arbitrary shapes, sizes or locations [11], which is computation intensive. In order to reduce computations, recent object detection approaches[12] use bottom-up saliency [13] to extract regions of probable objects in an image. Since these region proposals are category-independent, they still result in a large number of rectangular boxes that are irrelevant to the task. Task-specific top-down saliency approaches, such as the one proposed, can reduce computations significantly by extracting few task-relevant image regions. Being a pre-processor, their training and inference is faster compared to dedicated object detectors, and hence often computed with coarse spatial granularity.

Object localization is often used as a synonym for object detection[14]. However other approaches [15, 16] use the term to represent the process of identifying probable object locations, that need not be a pixel-accurate estimation as in object segmentation, but more accurate than a rectangular box estimated by object detection. In this paper we use the term 'object localization' to represent probable object locations. Our saliency map, which is a probability map that peaks at locations of a task defined object, can be used for object detection and segmentation applications.

In the proposed method, feature priors suited for the task of joint object localization and image classification, namely visual, spatial and neighborhood priors are used to compute saliency. We learn the visual and neighborhood priors based on SIFT [17] features through a conditional random field (CRF) [7], and the spatial prior through a spatial pyramid [18]. Fig.1 illustrates the idea of task-specific top-down salient object detection. Suppose the visual, spatial and neighborhood prior probabilities of features learnt for horse category are as shown in Fig.1 (a). This prior knowledge is used to estimate visual, spatial and neighborhood saliency separately for an image as shown in Fig.1 (b). The saliency values for yellow, green and red colored boxes are shown as bars with their corresponding colors. If a particular feature in a box has high prior probability of belonging to the horse category, then the box is assigned a high saliency. The yellow colored box is assigned with high visual saliency due to the high visual prior of the horse's head.

1.1. Combining top-down saliency and image classification

Based on the task at hand, task-specific top-down saliency approaches can be grouped into approaches for (i) image classification [10, 8, 19] and those for (ii) object localization [9, 7, 20]. In this paper, we demonstrate that the proposed saliency model is useful for both image classification and object localization tasks. The saliency model of [7] uses neighborhood and visual priors but lacks a spatial prior. On the other hand, the image classification approach of [18] has multi-scale spatial information, but lacks neighborhood prior. This paper is motivated by a need for top-down salient object detection and image classification joint framework that uses spatial, neighborhood and visual priors. A sparse coded spatial pyramid max-pooling (ScSPM) image classification can be improved by improving the discriminative quality of the dictionary, which can be obtained from the dictionary update of [7]. Similarly, saliency maps can be improved using the image classifier by leveraging information about presence of the object in an image. Thus, we propose an interconnected and mutually benefiting saliency-classification framework that helps reduce the computational cost.

Next, we briefly describe the main processes in our joint framework.

Category-aware sparse coding. ScSPM-based image classifiers are known to perform better [21] if the sparse codes are computed on a global dictionary whose atoms are representative features from all categories. Since the saliency models are developed based on sparse codes using individual object dictionaries, it would be computationally expensive to form a global dictionary
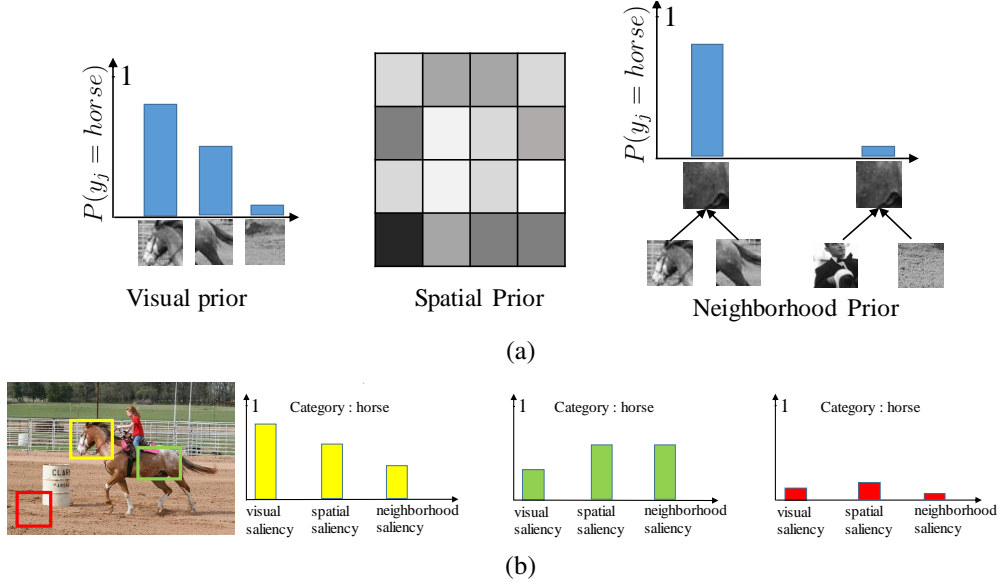
Figure 1: Priors for task-specific top-down saliency computation. (a) task-specific priors for horse category learnt from the training images. The visual, spatial and neighborhood saliency values for yellow, green and red colored image boxes in (b) are shown in their respective colors. The spatial prior is a 2-D distribution of horse patches in a 4×4 spatial grid with white indicating high probability and black indicating low probability. The horse's head (yellow box) has high correlation to the horse visual prior, resulting in large visual saliency. Similarly, it is less likely to find a horse patch at the position of the red box, resulting in lower spatial saliency.

using k-means clustering of thousands of patches from all categories and recompute sparse codes with respect to that dictionary. The proposed category-aware sparse coding utilizes the discriminative dictionaries already learned during saliency modeling, thereby enabling tight coupling between saliency models and the image classifier. This strategy helps in reducing the computational cost of feature coding for the classifier, while improving its classification accuracy.

Classifier-guided saliency model training. The image classifier used to update the saliency model is trained using a training set and validated using a validation set. The misclassified images during validation indicate that the features of those images are not represented adequately in the discriminative dictionary. The top-down saliency module is updated using such images. This classifier-guided saliency model training helps to improve not only the top-down saliency component, but also the image classification accuracy.

Saliency-weighted max-pooling. Conventional ScSPM image classifier [18] is blind to max-pooled vectors from the spatial pyramid blocks that contain an object and from those that do not contain any object. Hence, ScSPM often performs poorly in the presence of high background clutter. A novel saliency-weighted max-pooling proposed in this paper helps improve the performance. When a saliency model of a particular class is applied to an image, it highlights those patches that are likely to contain object parts belonging to that class. High saliency values appearing in a negative training image indicate false positive patches. Weighting corresponding max-pooled sparse codes with this high value and training the SVM [22] with a negative label will help the image classifier to label similar test images as negative.

Saliency refinement. The top-down saliency approaches of [7, 20] compute saliency of an
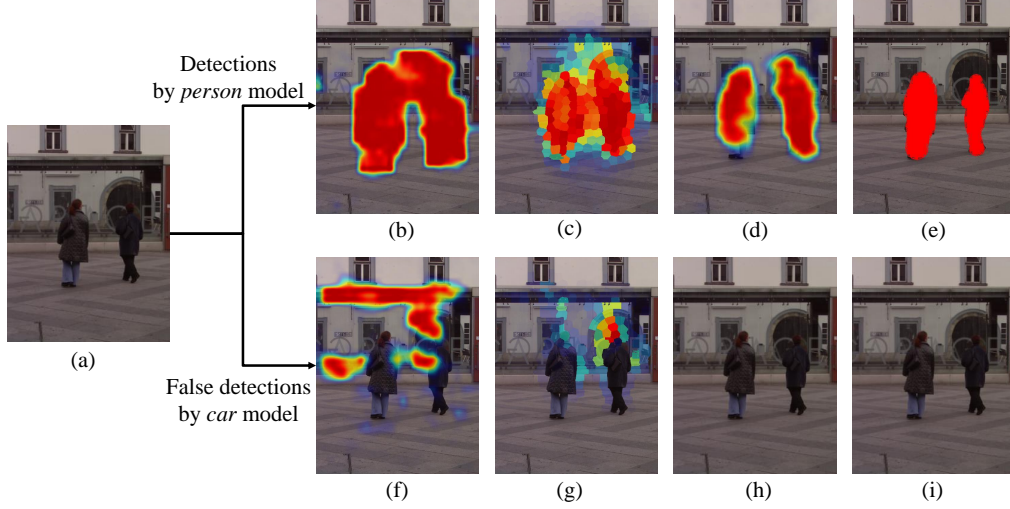
Figure 2: Comparison of saliency maps for an (a) input image by (b) Yang and Yang [7], (c) Kocak *et al.* [20], (d) proposed method and (e) ground truth on *person* model. False detections on applying *car* model of (f) Yang and Yang [7] and (g) Kocak *et al.* [20] are eliminated by (h) the proposed method resulting in a saliency map that is visually similar to (i) the ground truth.

image region without ascertaining whether the target object is present or not. Due to this, they often end up in producing false detection on negative images, which reduces the accuracy. In our framework, the image classifier is used to quantify the likelihood of the presence of an object and to refine the saliency map using a novel saliency refinement.

A preview of the effectiveness of our joint framework for top down saliency and image classification for the task of salient object detection is shown in Fig. 2. Fig. 2(a) shows an image containing two persons in a cluttered background. The person model of [7] and [20] are unable to distinguish between the persons due to their proximity in the image as seen in Fig. 2(b) and (c) respectively. Our framework produces a saliency map (Fig. 2(d)) closer to that of ground truth (Fig. 2(e)). Fig. 2(f) and (g) show the false detections by [7] and [20] respectively when their car saliency models are applied on the input image. As seen in Fig. 2(h), with the help of the classifier in our framework, we avoid these false detections.

In summary, the key idea of the proposed method is to add the image classification module both to update the saliency models, and to refine the saliency map for a given test image. On the other hand, the saliency inferred from the saliency models can also be used to improve the accuracy of the image classifier. The major contributions of this paper are the following:

1. A novel category-aware sparse coding strategy, which is computationally more efficient as compared to conventional sparse coding.
2. A novel framework to train saliency models with the help of a classifier.
3. A novel saliency-weighted image classification framework.
4. Refinement of saliency maps using image classifier confidence.

2. Related Work

First, we review the related works in task-specific top-down saliency estimation and image classification, which are the two major components of the proposed joint framework. Since the

usefulness of the saliency map obtained by the proposed approach is demonstrated for image segmentation and localization applications we briefly review algorithms for those applications also.

2.1. Task-specific top-down saliency

As mentioned in the previous section, task-specific top-down saliency approaches can be broadly categorized into approaches for (i) object localization and those for (ii) image classification.

2.1.1. Task-specific top-down saliency approaches for object localization

The locality-constrained linear coding (LLC) of SIFT features followed by max-pooling in a spatial context is used for saliency estimation in [23]. Even though they use the neighborhood prior through contextual max-pooling, it lacks a spatial prior. Khan and Tappan [24] use label and location-dependent smoothness constraint in a sparse code formulation to improve the pixel-level accuracy compared to the conventional sparse coding, but with additional computation cost.

Neighborhood prior of image patches on CRF is used for saliency estimation in [7]. The joint learning of CRF weights and dictionary was observed to improve the accuracy of the saliency map only marginally after 10 cycles, which makes the further learning of dictionary computationally less efficient. Moreover, this approach has limited ability to discriminate among the object categories. [20] improves on [7] by replacing patch-level SIFT features with RGB-based features from superpixels and by incorporating objectness [13]. Although this improved the accuracy of distinguishing objects from background, its ability to discriminate between object categories did not improve, causing large number of false detections if the test image contained irrelevant objects. By coupling saliency prediction with classification, such false detections are avoided in our framework. Also, using an image classifier to select the training images for [7], we could improve the saliency models, even with reduced training time.

2.1.2. Task-specific top-down saliency approaches for image classification

The appearance statistics of discriminative features from a pre-defined filter bank are used in [8] to distinguish different object classes. These salient features are learned using a weakly supervised approach, which are used for image classification. However, by only considering the image-level statistics of a feature and not its neighborhood information, they are unable to remove background patches leading to poor performance on datasets with heavy background clutter and viewpoint variations.

In [10], saliency is used to determine the discriminative patches for image classification. Saliency of a local region is calculated by considering its appearance along with its spatial location. Integrated max-margin learning is used to learn saliency and classifier. In [11], a random forest with discriminative decision tree is used to mine out the discriminative patches for fine-grained image classification. Category-specific color features are used in [19, 25] to modulate SIFT features. Shape features from the regions with higher color-attention are given more weight than those from lower attention. In [26], which is closest to our approach, a classifier is learned using randomly selected, random sized sub-windows from an image, which is then used to build and update a saliency map. Using this saliency map, the classifier samples more sub-windows in the salient region. The drawback here is that an error in the initial estimate of the saliency map gets propagated to consecutive iterations resulting in the failure of both classifier and saliency estimation. To avoid this, in our framework, training of top-down saliency model is separated

5

from the image classifier and they are integrated only after saliency reaches a certain level of accuracy.

2.2. Classification

Spatial Pyramid Matching (SPM) [27] has been widely used in image classification. In sparse coded SPM (ScSPM) [18], features were sparse coded and then max-pooled. This reduced the classifier complexity to $O(N)$ compared to $O(N^2\ or\ N^3)$ in SPM for N training images. Since max-pooled vectors from the spatial pyramid blocks that contain an object and those that do not contain any object are considered equally, ScSPM often performs poorly in the presence of high background clutter. We propose saliency-weighted max-pooling to improve this performance. The computational complexity of ScSPM is further reduced in LLC [28] by imposing feature locality constraint, and in [29] by compromising on the accuracy through division of the dictionary into cartesian product of sub-dictionaries. In [30], feature extraction from an automatically selected bounding box around objects is shown to improve image classification accuracy. However, this method needs iterative expansion of latent parameter space for effective localization of the object, which is computationally expensive.

2.3. Related Applications : Object localization and segmentation

In [15], bag-of-words representation is improved by spatial weighting of features using shape masks, causing foreground features to be boosted, thereby decreasing the influence of background clutter. High dimensional hypothesis clustering of shape mask is used for localization which requires separate annotation for each object within an image. i.e, if multiple objects from same category are present in an image, each of them needs to be labeled separately. Additionally, training the model requires images to be marked as *difficult* or *truncated*. The proposed method produces better localization result compared to [15] without the added requirement of such annotations.

Bag-of-feature based classification of local image regions having a fixed size is used in [16]. Absence of spatial consistency and context-type constraints causes many false detections leading to less accurate localization of objects. The class segmentation approach of [31] uses superpixels as the basic unit to build a classifier using histogram of local features within each superpixel. Histograms in a neighborhood are aggregated and used to regularize the classifier. A CRF built on superpixel graph further improved the segmentation accuracy. The performance of this model depends on the neighborhood size whose optimum size varies across object categories.

A fast and robust object segmentation proposed in [32] computes multi-class pixel-level object segmentation of an image through an integral linear classifier built on bag-of-words representation of local feature descriptors. Here, a large dictionary of 500,000 words is required and they use a cascaded classifier containing 2 or 3 linear classifiers. [33] uses bag-of-features for joint categorization and segmentation. The interaction between pixels and superpixels is modeled using random fields and global representation for categorization is achieved through a bag-of-features representation. The number of parameters to be learned for classification increases with increasing number of training images and number of categories. So, it is computationally infeasible to use this approach for large datasets like PASCAL VOC-07 [34]. The joint categorization and segmentation model proposed in [35] simultaneously learns segmentation, categorization, and dictionary learning parameters. Simplicity of bag-of-features representation limits the performance of both.

3. The proposed joint framework

3.1. Brief review on top-down saliency of [7]

In [7], dense SIFT features are extracted from regular, rectangular gray-scale image patches and their sparse representations are initially computed with a dictionary D_n formed by k-means clustering. The dictionary is formed from cluster centroids and it represents the most representative patches of an object category. The use of sparse coding of SIFT features results in a more compact and discriminative representation which helps to model feature selectivity for saliency map. Using these sparse codes as latent variables, a conditional random field is learned. The CRF node weight w_n is initialized with a binary SVM classifier weight learned on these sparse codes and pairwise energy is set to 1. The dictionary and CRF weights are jointly learned in 20 iterations using max-margin framework. Finally, loopy belief propagation is used to infer saliency values on test images. In the proposed method, the number of iterations is set to 10 to save computations because the improvement in accuracy of the model after 10 iterations is insignificant compared to its computation cost.

One major drawback of [7] is that its focus is on distinguishing objects from background and not from other objects, resulting in a large number of false positives as shown in Fig. 2(f). Our method overcomes this limitation by integrating a classifier that is trained on novel category-aware sparse codes, which also results in considerable savings in computation, especially when dealing with large datasets.

3.2. Brief review on ScSPM image classification [18]

In a ScSPM-based classifier [18], dense SIFT features are extracted from gray-scale image patches. SIFT features from training images are used to form a global dictionary. The SIFT features of an image are sparse coded using this dictionary. The spatial distribution of the features in the image is encoded in the max-pooled image vector through a multi-scale max-pooling operation of the sparse codes on a 3-level spatial pyramid [27]. The max pooled feature is more robust to local transformations than mean statistics in histogram. Biophysical evidence in visual cortex (V1) [36] also establishes the use of max-pooling. Image-label pairs of training images are used to train a linear binary SVM classifier.

3.3. System Overview

Fig. 3 shows an overview of the proposed joint framework for salient object detection and image classification. Similar to the original framework of [7] and [18], we only use SIFT features extracted from gray-scale images. From every image I^i, image patches with a fixed size and grid spacing are extracted. For each patch j, its dense SIFT feature $f^{i,j}$ and ground-truth $y_n^{i,j} \in \{-1, 1\}$ are computed, where -1 and 1 denote the absence or presence, respectively, of object n in patch j. We split the training images (Train1+Train2) into Train1 (training set) and Train2 (validation set), to save computations by avoiding the update of saliency model with a training image whose features are already well represented in the saliency model. The discriminative dictionary D_n for each object category n is initialized with the centroids of the clusters formed by k-means clustering applied on positive SIFT features.

Similar to [7], the sparse codes of SIFT features with D_n are used as latent variables in our saliency models. The sparse codes of positive and negative patches from Train1 images are used to learn a linear SVM and the SVM weights are used to initialize the the CRF node weight w_n. The pair-wise energy is set to 1 as in [7]. Following [7], for each category n, the dictionary D_n
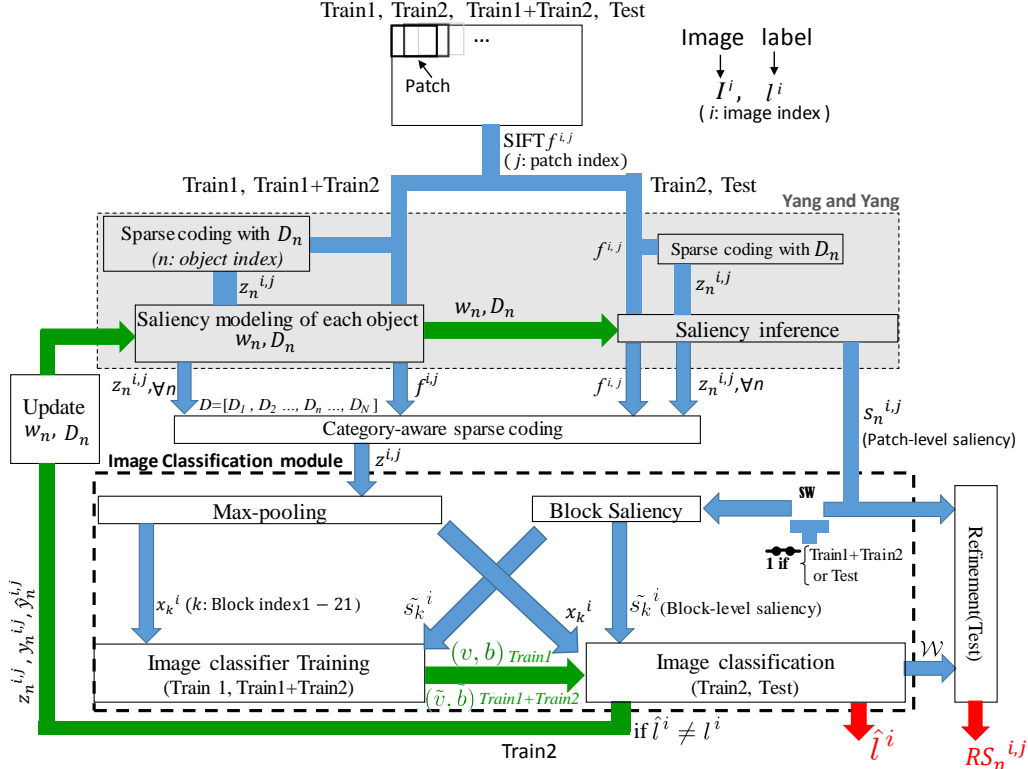
Figure 3: Overview of the proposed framework for classifier-guided salient object detection. The shaded region represents the framework of [7]. **SW** is connected only during training and testing of saliency-weighted classifier. **SW**=1 and **SW**=0 indicate that the switch is connected and disconnected, respectively. Green arrows indicate output of training stages and red arrows indicate the final saliency map and classifier output on a test image.

and the CRF weight w_n are jointly updated using Train1 images for 10 iterations. The updated dictionary D_n represents the most representative patches of the nth object category, and the CRF weights learnt on sparse codes represents our saliency model. Since sparse code is used as a latent variable for saliency modeling, the sparse code $z_n^{i,j}$ of $f^{i,j}$ with D_n, $n \in \{1, 2, ..., N\}$, is recomputed in each of these 10 iterations.

Following 10 iterations, the *classifier-guided saliency model update* process is iterated 2 times. An image classifier that uses D_n and $z_n^{i,j}$ of all N object categories, and trained on Train1 images is used to choose images from Train2 to update saliency models. The image classifier performs better if the sparse representations are on a global dictionary D than N separate sparse representations on N discriminative dictionaries D_n[21]. So, we propose a *category-aware sparse coding* strategy that reuses each D_n and $z_n^{i,j}$ computed during saliency estimation for N categories to produce a global representation $z^{i,j}$ of $f^{i,j}$. This category-aware sparse representation helps to reduce the additional computational requirement for image classification compared to the conventional sparse coding with D [18].

The category-aware representation of all patches in image i are max-pooled over a multi-scale spatial pyramid and the max-pooled vector in each block k of the spatial pyramid is represented

as $x_k{}^i$. The max-pooled vectors from all 21 blocks are vertically concatenated followed by l_2-normalization to form the max-pooled image vector x^i. The x^i from all images in Train1 and their corresponding image labels l^i are used to train a linear SVM with weight v and bias b (**SW** = 0 in Fig.3). The classifier is used to predict the label $\hat{l}^i$ of Train2 images. A misclassification $\hat{l}^i \neq l^i$ is an indication that the corresponding object model has not been learned by the CRF comprehensively enough to include its appearance as in the misclassified image. Thus, the classifier selects those images with which the saliency model w_n needs to be updated. Consequently, the corresponding object dictionary D_n is also updated, which, in turn, refines the global dictionary D formed by concatenating the object dictionaries. We use max-margin approach to identify the most violated constraints for the misclassified image and to update D_n and w_n accordingly (classifier-guided saliency update, **SW** = 0 in Fig.3).

After the above mentioned *classifier-guided saliency update*, we improve the image classifier using saliency maps. The proposed saliency-weighted max-pooling operation (**SW**=1 in Fig.3) weights the max-pooled vectors $x_k{}^i$ of each block k with its block saliency $\tilde{s}_k{}^i$. We infer the saliency maps from all training images Train1+Train2 and compute their saliency-weighted max-pooled $\tilde{x}^i$ vectors, which are used to train a linear SVM $(\tilde{v}, \tilde{b})$. This *saliency-weighted image classifier* $(\tilde{v}, \tilde{b})$ is applied on a test image to refine the saliency map, and to classify it. At the end of the training stage, we get (i) a saliency model with updated CRF weight w_n and dictionary D_n and (ii) a saliency-weighted image classifier $(\tilde{v}, \tilde{b})$.

For test images, the saliency of each category, $s_n^{i,j}$ is estimated by CRF inference using loopy belief propagation as in [7]. These saliency maps are refined using the posterior probability $\mathcal{W}$ estimated from the saliency-weighted classifier as explained in Sec. 6. This *refined saliency $RS_n^{i,j}$* reduces the false detections in the saliency as shown in Fig. 2(h) and supported quantitatively in our experimental results. In summary, the proposed approach is able to simultaneously identify and localize the object categories present in a test image.

4. Category-aware sparse coding

The category-aware sparse coding reuses the object dictionary D_n and corresponding spare codes $z_n^{i,j}$ computed by the saliency component. The dictionary update in the saliency component improves the discriminative quality of the object dictionary D_n, and a global dictionary D formed by concatenating updated object dictionaries of all categories helps improve the image classification performance. Moreover, this approach reduces the additional computations incurred by forming a global dictionary D using k-means clustering of thousands of patches from all categories followed by recomputation of sparse codes with respect to that dictionary. Spatial pyramid max-pooling of the category-aware sparse codes (ScSPM) are used in our classification module.

Tight coupling between saliency modeling and image classification is obtained through the proposed category-aware sparse coding. It is to be noted that the following discussion pertains to computation of category-aware sparse code $z^{i,j}$ of a feature $f^{i,j}$. For simplicity, we drop the superscripts i and j in this section. Conventional sparse representation of a given feature f aims to achieve the minimum reconstruction error while using sparse number of atoms from D, i.e.,

$$z = \arg\min_z \|f - Dz\|_2 + \lambda \|z\|_1 . \tag{1}$$

Since the objective of our image classifier is to improve saliency estimation, we introduce a category-aware constraint to the feature coding for classification, resulting in a novel category-aware sparse code z. i.e., the category-aware sparse code z aims to achieve minimum reconstruc-
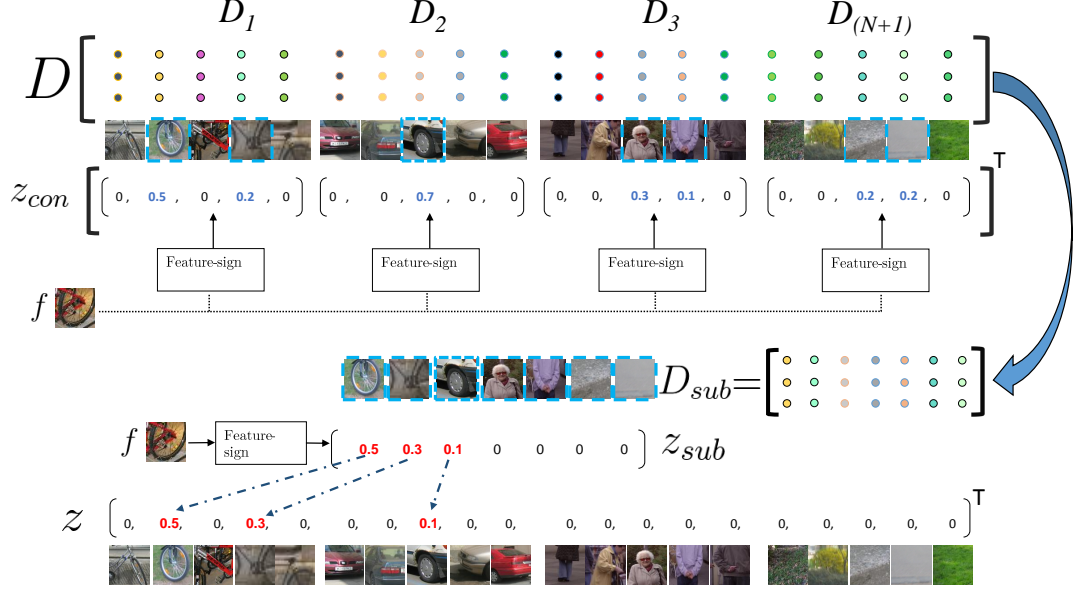
Figure 4: Illustration of category-aware sparse coding for classification (best viewed in color).

tion error while representing the feature f with respect to each category dictionary D_n as well as to the global dictionary D.

4.1. Formulation

Let D be the global dictionary formed by concatenating the individual object dictionaries, i.e. D has a structure $[D_1, D_2, \ldots D_{(N+1)}]$, where D_n represents each object dictionary and $(N + 1)$ includes the N object categories and the background class. The saliency model for an object category n and its corresponding dictionary D_n is learned as described in [7]. The dictionary for the background class is formed by k-means clustering of background patches; thus, background features also have a sparse representation. The size of D is $k \times r_D$ where $r_D = (N \cdot r + r_{bg})$, k is the dimension of the feature vector, r and r_{bg} are the number of atoms in the dictionary for each object class and the background, respectively.

The objective function for category-aware sparse coding is

$$z = \arg\min_z \|f - Dz\|_2 + \lambda_1 \|z\|_1 + \lambda_2 \sum_{n=1}^{N} (\|f - DC_n z\|_2 + \lambda_3 \|C_n z\|_1), \qquad (2)$$

where the first two terms are the conventional sparse coding of feature f with l_1 constraint and the third term imposes our category-aware constraint.

C_n is a selection matrix that selects the atoms of a particular object category n from D. C_n is derived from a zero matrix of size $r_D \times r_D$ by replacing its mth diagonal element with 1, if the mth atom in the dictionary D belongs to the category n. For example if D contains 6 atoms ($r_D = 6$) with the third and fourth atoms of D belonging to category n, then

10

$$C_n = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix},$$

and $C_n z$ selects the elements of z that belongs to category n. $\|f - DC_n z\|_2$ represents the reconstruction error for the sparse representation of feature f with category dictionary D_n (non-zero elements of $DC_n = D_n$) and the l_1-norm constraint $\|C_n z\|_1$ ensures that only few atoms of D_n is used in this sparse representation.

4.2. Approximate solution

Since a closed-form solution of Eq. (2) is not possible, we propose a computationally efficient approximate solution. Since the sparse codes z_n of each feature f with respect to object dictionaries D_n are already available from the saliency module, we develop a strategy to reuse these codes which is computationally more efficient than conventional sparse coding of the feature f with global dictionary D.

The dictionary formation process and the category-aware sparse code vector z are shown in Fig. 4. The proposed solution minimizes Eq. (2) in two steps. First, the sparse codes z_n, $\forall n \in \{1, ...N\}$ of each feature f with all category dictionaries D_n $\forall n \in \{1, ...N\}$ are computed using feature-sign solver [37], which reduces the third term in Eq. (2), i.e $\lambda_2 \sum_{n=1}^{N} (\|f - DC_n z\|_2 + \lambda_3 \|C_n z\|_1)$.

Let z_n be the sparse code for feature f evaluated with respect to object dictionary D_n. We form a vector z_{con} by vertically concatenating the sparse codes of f obtained for each D_n as shown in Fig. 4. The non-zero terms in z_{con} point to atoms in D that contribute to minimizing the reconstruction error of f and hence, can effectively represent it. We pick these atoms from D to form the classifier dictionary D_{sub} and generate sparse code vector z_{sub} for f on D_{sub} to minimize the overall objective function.

Since the number of non-zero terms in z_{con} is small, the number of atoms in D_{sub} is much lesser than in D resulting in lesser computation for generating z_{sub}. Since the number of atoms in D_{sub} is different for every feature and the ScSPM classifier needs a dictionary of the same size as D, we need to represent z_{sub} in a vector with the same length as z_{con}. Since D_{sub} is formed by picking atoms from D, the elements of z_{sub} can be placed in their respective locations of the category-aware sparse code z having same length as the number of atoms in D, initialized with a zero vector. Let a sparse code element z_{sub_p} in the pth position in the code vector z_{sub} correspond to atom d_p in D_{sub}. If d_p is the mth atom in D, then the code z_{sub_p} is placed in the mth position of category-aware sparse code z. The number of atoms in D_{sub} for PASCAL VOC 2007 dataset [34] ranges from 300-400, which is much smaller compared to a typical classifier dictionary size for the same dataset, which ranges between 8000 and 12000 [30].

4.3. Computational Complexity

We use the Feature Sign sparse solver (FS) [37] whose time complexity to encode feature f is $O(Lk) + O(LT_f)$ when the dictionary size is $k \times L$ and T_f is the sparsity of the code (number of non-zero elements in the code). Using FS solver on the global dictionary D (obtained by concatenating category dictionaries) will result in a computational complexity of $O(r_D k) + O(r_D T_f)$,

11

where $r_D = N \cdot r + r_{bg}$, which is very large for datasets with a large number of classes (N). One possible approach to reduce the computation is to reduce r_D by forming a smaller global dictionary by clustering of features from all categories (instead of concatenating category dictionaries). However, it may result in loss of fine-grained information that helps in distinguishing similar classes.

In the proposed framework, for every feature, FS solver is used in two stages - first, during saliency estimation, with respect to dictionaries D_n and then with respect to sub-dictionary D_{sub}. Since there is no dependency between the first stage solvers, a parallel implementation can effectively result in a time complexity of $O(rk) + O(rc_n)$ per feature for each object class where c_n is the sparsity of the sparse code with D_n. Since r_D is nearly $(N + 1)$ times larger as compared to r, parallel implementation of proposed framework is $(N + 1)$ times less complex as compared to the time complexity of conventional sparse coding on D. For the same sparse penalty λ, we have observed that c_n in each category code is less than T_f, the sparsity with the global dictionary D. For sparse coding using D_{sub}, each feature requires an additional time complexity of $O(r_{sub}k) + O(r_{sub}s_{sub})$, where s_{sub} is the sparsity of z_{sub}. Since sparse codes are already available from the saliency estimation stage, the second round of sparse coding computations are the only additional computations required for sparse coding of classification, resulting in significant savings in computation.

5. The image classification module

Classifiers are used for saliency model training, saliency map refinement and for image classification. The proposed framework tightly couples these processes.

5.1. Classifier to train saliency model (classifier feedback)

The classifier uses 3-level spatial pyramid max-pooling [18] of category-aware sparse codes. i.e, an image i is divided into 21 blocks. Each block k is represented by a single max-pooled vector x_k^i of dimension $(N \cdot r + r_{bg}) \times 1$ formed by element-wise maximum of the category-aware sparse code vectors $z^{i,j}$ in that block. Each image is represented with a max-pooled vector x^i of dimension $21(N \cdot r + r_{bg}) \times 1$, formed by vertical concatenation of the max-pooled vectors from each block.

Let $\{\mathbf{x}^i, l^i\}$, $i \in$ Train1 be the training data where x^i is the l_2-normalized vector from image i and $l^i \in \{1, -1\}$ indicates the presence or absence of the target object in that image i. A one-vs-rest (binary) linear SVM classifier with SVM weight v and bias b is trained on half of the training images Train1 by minimizing following objective function [22]

$$\arg\min_v \|v\|^2 + C \sum_{i \in \text{Train1}} max\,(0, 1 - l^i(v^\top x^i + b)\,), \tag{3}$$

where C is the cost of constraints violation.

The max-pooled vector of each image i from the other half of the training set, Train2, are used to validate the classifier using $f(x^i) = v^\top x^i + b$. Correct classification of image i is indicated by $(f(x^i) \cdot l^i) > 0$. The saliency model corresponding to the misclassified object is updated through refinement of CRF weights and dictionary (classifier feedback). For example, if a bike image (n=bike) is misclassified by the bike classifier, the bike saliency model is updated using this image. Let $Z_n^i = [z_n^{i,1}, z_n^{i,2} \ldots z_n^{i,t}]$ be the set of all sparse codes of a misclassified image i using D_n. and $Y_n^i = [y_n^{i,1}, y_n^{i,2}, \ldots y_n^{i,t}]$, $y_n^{i,t} \in \{-1, 1\}$ be the ground truth label for target n presence

12

in each patch. Here t is the total number of patches in the image. If $\hat{Y}_n^i$ are the labels predicted using category n CRF model, the loss function for the image is [7]

$$\beta(w_n, D_n) = E(\hat{Y}_n^i, Z_n^i, w_n) - E(Y_n^i, Z_n^i, w_n), \tag{4}$$

where E is the energy function of CRF built on a four connected graph conditioned on sparse codes Z_n^i (for details please refer to [7]). The ground truth energy $E(Y_n^i, Z_n^i, w_n)$ is less than any other energies $E(Y, Z_n^i, w_n)$ by a large margin $\Delta(Y, Y_n^i)$. i.e, $E(Y_n^i, Z_n^i, w_n) \le E(Y, Z_n^i, w_n) - \Delta(Y, Y_n^i)$ [7]. Here Y represents set of binary labels assigned to patches in the the image i and $\Delta(Y, Y_n^i) = \sum_{j=1}^t \mathbb{I}(y^j, y_n^{i,j})$ indicates the margin function, where $\mathbb{I}$ is an indicator function which is 1 when $y^j \neq y_n^{i,j}$ and 0 otherwise. The most violated constraints are identified by solving

$$\hat{Y}_n^i = \arg\min_Y E(Y, Z_n^i, w_n) - \Delta(Y, Y_n^i), \tag{5}$$

which is used to update the CRF weights w for object o as [7],

$$w_n = w_n - \rho_0 \frac{\partial \beta}{\partial w_n}, \tag{6}$$

where $\rho_0 = 10^{-3}$ is the learning rate.

Following [7], the updated CRF weight is used to update the object dictionary D_n for a given object n (e.g bike) as

$$D_n = D_n + \rho_0 \frac{\partial \beta}{\partial D_n}. \tag{7}$$

5.2. Saliency-weighted classifier

Once the object dictionary refinement and saliency model learning is complete, we use saliency maps to improve classifier accuracy. Fig. 6 illustrates the saliency-weighted classifier pipeline. The max-pooled vector at block k, x_k^i is weighted with the corresponding block-saliency value $\tilde{s}_k^i$ to form the saliency-weighted max-pooled vector $\tilde{x}_k^i$ for that block. The saliency-weighted max-pooled vectors from all 21 blocks of the spatial pyramid are vertically concatenated to form the saliency-weighted image vector $\tilde{x}^i$. An image classifier is learned to indicate the presence or absence of target object ($l^i \in \{1, -1\}$) using $\tilde{x}^i$ from all the training images, i.e (Train1+Train2).

5.2.1. Block saliency computation

The saliency model of a particular class highlights those patches that are likely to contain object parts belonging to that class. High saliency values indicate either object regions in a positive image or possible false positive patches in a negative training image. Weighting corresponding max-pooled sparse codes with this high value and training the SVM with its corresponding image label will help the image classifier to reduce false detections and improve its performance against background clutter. To this end, for each image, our objective is to determine a saliency weight for the max-pooled vector in each of the 21 spatial blocks of the pyramid, using the N saliency maps computed for the N categories.

The first step involves finding the saliency value for each category in each block of the spatial pyramid. Choosing the maximum saliency for each block would lead to a poor representation caused by outliers in the saliency map. Using the mean saliency of an object within a block may

Figure 5 (a):

$$\mu^k_{car} = (0.9+0.7+0.6)/3 = \mathbf{0.733} \quad (\nu=3)$$

Mean of Top ν values $\Rightarrow \mu^k_{car}$

Saliency values in block: 0.1, **0.7**, **0.6**, 0.3, **0.9**, 0.4, 0, 0, 0

Figure 5 (b):

μ_{car}

0	0	0	0
0.1	0.3	0.6	0.1
0.2	0.8	0.73	0.1
0	0	0	0

$+$

μ_{bike}

0	0	0	0
0	0	0.3	0.1
0	0	0.2	0.2
0	0	0	0

$+$

μ_{person}

0	0	0	0
0	0.3	0.2	0
0	0.2	0	0
0	0	0	0

$=$

$\tilde{S}^i$

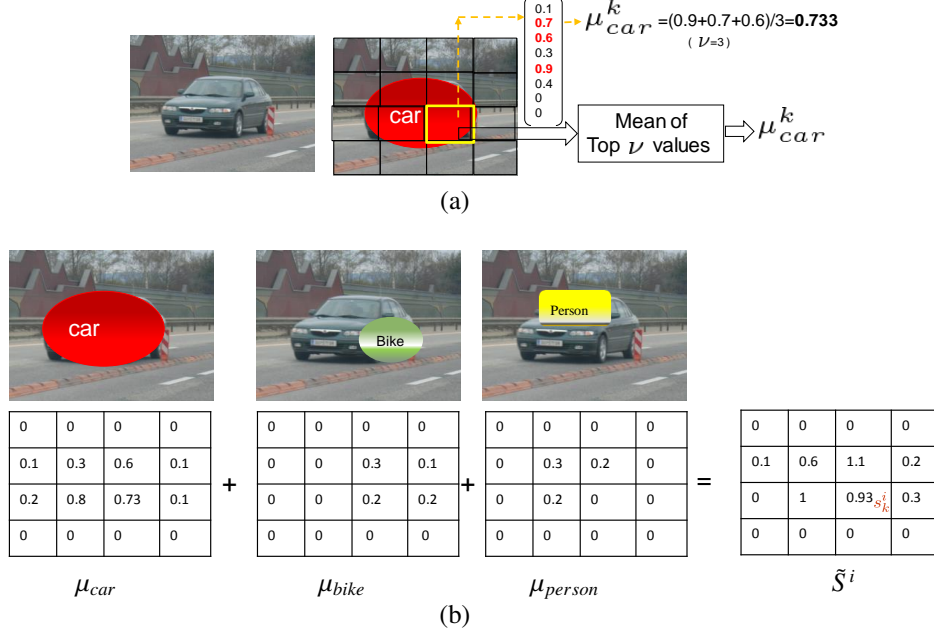| 0	0	0	0
0.1	0.6	1.1	0.2
0	1	0.93 ($\tilde{s}^i_k$)	0.3
0	0	0	0

Figure 5: Illustration of block saliency computation for blocks of spatial pyramid (best viewed in color).

result in loss of saliency information at higher levels of the pyramid. This is attributed to the presence of many background pixels with low saliency values that are present in the larger image areas covered by these levels. Hence, we use the mean, μ, of the top ν, ($\nu = 2$) saliency values of an object category in a block as shown in Fig. 5(a).

The block saliency $\tilde{s}_k^{\ i}$, which is the weight of the max-pooled sparse code vector for the kth block is obtained as $\tilde{s}_k^{\ i} = \sum_{n=1}^{N} \mu_n^k$ (Fig. 5(b)). Since we use mean of top saliency values in a block corresponding to all object models, background patches are suppressed compared to the object patches.

5.2.2. Saliency-weighted max-pooling

Our weighting criteria is simple and computationally efficient enabling its use in larger datasets having multiple objects in an image. Every element of the max-pooled vector from a block k is multiplied with its corresponding block saliency $\tilde{s}_k^i$ to form the saliency-weighted max-pooled vector $\tilde{x}_k^i$ for that block. i.e

$$\tilde{x}_k^i = \tilde{s}_k^i \cdot x^i_k, \quad \forall k \in \{1, 2, \ldots 21\} \tag{8}$$

The l_2-normalized saliency-weighted max-pooled vectors $\tilde{x}_k^i$ from entire training set (Train1+Train2) are used to learn a one-vs-rest (binary) linear SVM classifier $(\tilde{v}, \tilde{b})$ for image classification as in Eq. (3) [22].

$$\arg\min_{\tilde{v}} \|\tilde{v}\|^2 + C \sum_{i \in (\text{Train1}+\text{Train2})} max\,(0,\ 1 - l^i(\tilde{v}^\top \tilde{x}^i + \tilde{b})\,), \tag{9}$$
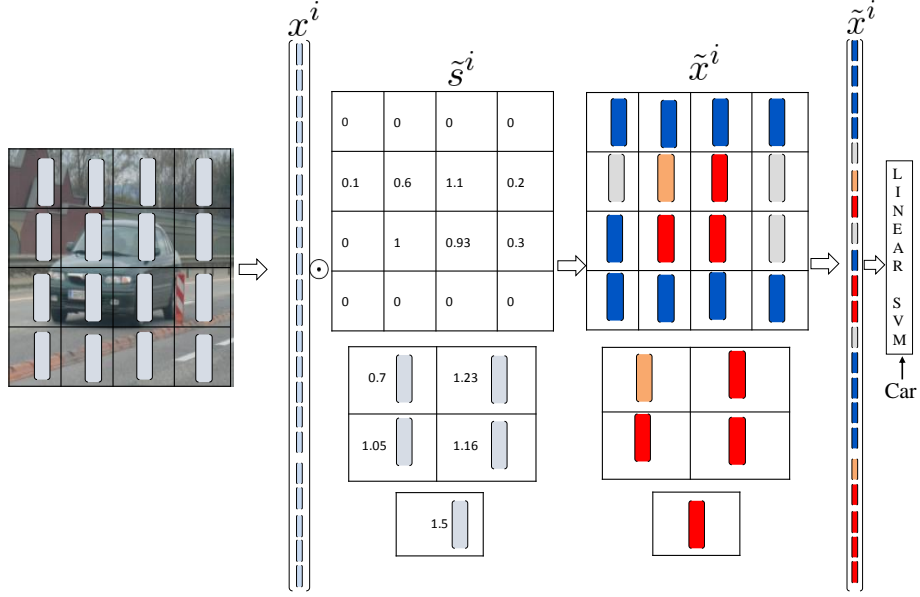
14

Figure 6: Illustration of saliency-weighted max-pooling (best viewed in color).

6. Saliency inference and refinement

The final saliency map generated by the proposed method is a weighted version of the one proposed in [7] where the weights are based on the output of the saliency-weighted classifier explained in Sec. 5.2. In [7], a four connected graph having Markov property is formed on image patches based on their spatial adjacency. The probability of label $y_n^{i,j} \in \{-1, 1\}$ indicating the absence or presence of object n respectively at a patch j is computed from its neighbours using marginal probability [7]

$$P(y_n^{i,j} \mid z_n^{i,j}, w_n) = \sum_{y_n^{i,\mathcal{N}(j)}} p(y_n^{i,j}, y_n^{i,\mathcal{N}(j)} \mid z_n^{i,j}, w_n), \tag{10}$$

where $z_n^{i,j}$ is the sparse code of a patch j, w_n is the CRF weight vector and $\mathcal{N}(j)$ is the neighbourhood of node j with label $y_n^{i,\mathcal{N}(j)}$.

Saliency of a patch j is given by

$$s_n^{i,j} = P(y_n^{i,j} = 1 | z_n^{i,j}, w_n), \tag{11}$$

which is inferred using loopy belief propagation [7]. The neighborhood and visual priors of SIFT features are used in this saliency computation. The refined saliency $RS_n^{i,j}$ for patch j is

$$RS_n^{i,j} = s_n^{i,j} \cdot \mathcal{W}, \tag{12}$$

where $\mathcal{W}$ is derived from SVM confidence ($\tilde{v}^\top \tilde{x}^i + \tilde{b}$) of the saliency-weighted classifier. Since the weight is the same for all patches, it is equivalent to multiplying the saliency map of the image

15

with the weight. The spatial prior is used by the saliency-weighted classifier for the computation of $\mathcal{W}$ through its spatial pyramid max-pooling operation. $\mathcal{W}$ is assigned a high value if the location of the object of interest in a test image matches its spatial prior (object location in the training data). Objects appearing at spatial locations which are significantly different from their training data are assigned lower $\mathcal{W}$ values.

The training images of a dataset indicate the maximum number of object categories in an image. If there is only 1 object category per image and the class with highest SVM confidence matches the model being tested, $\mathcal{W} = 1$ and the generated map becomes the final saliency map. If there are multiple object categories per image, we evaluate the classifier confidence for each of the object categories within a test image and sort them in descending order. If the confidence of an desired object category is ranked below the maximum number of objects per image in the training set, it is highly unlikely that object is present in the image and hence the refined saliency map will have no salient patches in that test image. If the confidence of the object category is ranked within the highest number number of objects per image, a rescaled classifier confidence serves as $\mathcal{W}$. In Graz-02 dataset [38], which has only one object class per image, the saliency map corresponding to the object class predicted by the saliency-weighted classifier is weighted by 1; the saliency maps of the other categories are weighted by 0. While this strategy of weighting the saliency map might appear to be totally dependent on the classifier performance, it must be noted that the classifier uses saliency-weighted pooling so that an accurate saliency map will contribute to lowering the error in classification. In PASCAL VOC-07 which has 20 classes, there is no training image with more than 5 object classes. If the classifier confidence for a particular class is ranked more than 5, $\mathcal{W}=0$. Since most of the training images have 1 or 2 object classes, $\mathcal{W}=1$ if the classifier confidence is ranked first or second. For the remaining positions, $\mathcal{W}$ is taken as the linearly scaled classifier confidence.

7. Experimental Results

We evaluated the performance of our joint framework on Graz-02 and PASCAL VOC 2007 datasets. SIFT features are extracted from 64×64 patches with a grid spacing of 16 pixels as in [7]. We choose λ in the sparse code formulation $z_n = \arg\min_{z_n} \|f - D_n z_n\|_2 + \lambda \|z_n\|_1$ to be 0.15 for both classifier and saliency modules. We evaluate top-down saliency using precision rates at equal error rate (EER). The saliency maps are thresholded at 100 levels between 0 and 1 and a precision recall graph is drawn. The EER refers to the point at which the precision is equal to the recall.

7.1. Training and testing image selection
Graz-02 dataset

Graz-02 has 3 object categories- bike, car and person, each having 300 images with pixel-level object annotations and an additional 65, 120 and 11 images without object annotations in each category respectively. Apart from this 380 background images are also present.
We conduct the experiments on different sets of training and test image combinations. For comparison of the proposed top-down saliency with recent top-down saliency approaches, we only consider object annotated images for training and testing. Following [7, 20], odd numbered images are used for training the proposed saliency model and even numbered images for testing.
Secondly, to compare classifier performance with related image classifiers, the same training and test images selection procedure in Bilen *et al.* [30] is used. All (1096) object images are used for

classification. For each category, 150 training images are selected at random, and remaining are used for testing. The average results of 10 such experiments are reported.

PASCAL VOC-07 dataset

PASCAL VOC 2007 is a challenging dataset consisting of 20 different categories with some images having objects from multiple object categories. Similar to [7], we evaluated the performance of our saliency model on 210 segmentation annotated test images. Since there are only 422 segmentation images to train 20 categories, we use additional images from the object detection challenge as in [7] (nearly 150 images per category). For comparison with related classifier approaches, the training, validation, and test image combinations in the *classification challenge* of the dataset is used.

7.2. Top-down saliency

7.2.1. Graz-02 dataset

For each object category, an object dictionary with 512 atoms and corresponding CRF parameters are learned for 10 iterations. From $3 \times 512 = 1536$ object atoms and 512 background atoms, a global dictionary of 2048 atoms is formed by concatenation. The ground truth label of a patch is 1 if at least 25% of its pixel belongs to object of interest, and -1 otherwise. In [7], training for each category is done using 150 positive images and 150 background images (column T1 in Table 1). To improve performance against the false positive detection, training of proposed framework uses negative images from other categories as well (T2 in table. 1). Since T2 has 150 positive images and 450 negative images, the training set to re-train [7] is balanced by randomly selecting 150 negative images from 450 negative images available in T2.

Since the proposed framework requires training and validation cycles, T2 is divided into T2a (training set) and T2b (validation set). T2a contains 70 positive images for each category and 80 negative images, 30 of which are from background and 25 each from the other 2 categories. Since 70 positive images from each category are used to form T2a, the remaining 80 images per category form 320 images in T2b. Misclassified images from T2b are used to update the top-down saliency model.

Table 2(a) compares the patch-level precision rates at EER of the proposed saliency detection with [7] on the Graz-02 dataset. In [7], the authors tested each saliency model on object annotated images from its respective category and background. An ideal top-down saliency estimator should be able to distinguish the object from background clutter as well as from other objects. So, we tested [7] on images from all categories in the dataset, i.e, we use their model trained on T1 and evaluated on all 600 images. The average EER of 54.9% is less than 73.7% reported in [7]. Since our training mechanism involves negative images from other categories in addition to the background category, we used T2 to train [7]. The increase of about 5% in EER illustrates the utility of negative images for training.

Effect of saliency refinement. Our saliency modeling process has 2 stages - in the first 10 iterations, the models are learned using T2a and in the second stage the classifier feedback improves the models through misclassified images. In this experiment, we consider only the first stage and demonstrate the utility of refining the saliency map in table 2(a) (column 3). The saliency models trained using T2a for 10 iterations resulted in a mean precision rate at EER of 54.1% without saliency refinement and without classifier feedback. Using saliency model and dictionaries obtained at this 10th iteration of saliency modeling, we train a saliency-weighted image classifier (Sec. 5.2) that uses multi-class SVM [39] on the entire training images. On test

Table 1: Graz-02: Saliency training sets used in our experiments.

Image set	T1	T2	T2a	T2b
Number of positive training images	150	150	70	80
Number of background images	150	150	30	80
Number of negative images except background	0	2x150	2x25	2x80
Total number of training images per category	300	600	150	320

Table 2: Precision rates at EER (%) of proposed method against other top-down saliency approaches on all (600) test images of Graz-02 dataset.

(a) **Patch-level**

Algorithm	Yang and Yang [7]	Yang and Yang [7]	Proposed method	Proposed method	Proposed method on a classifier of 100% accuracy
Training Set	T1	T2	T2a	T2	T2
Number of trg. iter.	20	20	10	10+2	10+2
Bike	62.5	69.4	**75.6**	**75.6**	79
Car	53.6	53.2	53.8	**58.3**	65.8
Person	48.6	57.2	62.8	**64.5**	71
Mean	54.9	59.93	64.06	**66.13**	71.9

(b) **Pixel-level**

Algorithm	Yang and Yang [7]	Kocak *et al.* [20]	Proposed method
Training Set	T2	T2	T2
Number of trg. iter.	20	20	10+2
Bike	59.43	59.92	**64.4**
Car	47.36	45.18	**50.9**
Person	49.82	51.52	**56.4**
Mean	52.2	52.21	**57.23**

images, the saliency maps are weighted using classifier confidence output for that image as described in Sec. 6. There is a gain of 10% in precision rates at EER due to the saliency refinement and the resulting model outperforms [7] with a 5% gain in precision rates at EER. Moreover, we use only 150 training images and 10 iterations when compared to 300 training images and 20 iterations in [7] for saliency modeling. The selection of relevant atoms from the dictionary that can represent the feature for its sparse representation and a saliency-weighted classifier trained on the category-aware sparse codes jointly contribute to improve the performance.

The bicycle model causes few false detections on the *car* image of Fig. 8, due to the similarity in the structure. For example, the round shaped tyre patches in *car* is detected due to the similarity with the bicycle tyre. Since the image classifier uses high level information in the image, it could better predict that bicycle is absent in the given image and it removed the false detections as shown in Fig. 8 (c). Similarly, the shadow of human in the *background* image introduced false detections while inferring a person model. Again, classifier-based saliency refinement removed those false detections.

Table 3: Precision rates at EER (%) of proposed method against other localization approaches on 150 test images of Graz-02 dataset.

Algorithm	Fulkerson *et al.* [16]	Marszalek and Schmid [15]	Proposed method
Bike	66.4	61.8	**67.3**
Car	54.7	53.8	**59.8**
Person	47.1	44.1	**57.1**
Mean	56.07	53.23	**61.4**

Figure 7: Qualitiative comparison of saliency maps of Yang and Yang [7] and Kocak *et al.*[20] with the proposed method.

Failure cases of the saliency refinement. Saliency refinement largely depends on the accuracy of the image classifier. It may introduce two types of errors in the saliency map, (i) false negative, when the image classifier wrongly predicts the absence ($W = 0$) of the object in a positive image Fig.9 (top row) (ii) false positive, when the image classifier wrongly predicts the object presence in a negative image Fig.9 (bottom row). Due to viewpoint changes, the image classifier fails to predict bicycle presence in the first case and in the second image, shirt hanging on the door matches with the position and features of person class. However, there are very few such errors in our model.

Performance on 100% accurate image classifier. A 100% accurate image classifier will assign $W = 1$ for positive test images and $W = 0$ for negative test images. To evaluate the performance under 100% accuracy, we manually assigned these values to W, based on the ground-truth of the test image. Such an image classifier with 100% accuracy can remove both these errors and produce a better saliency map as shown in Fig.9 (c). Such a classifier improves our saliency accuracy on 600 test images to 71.9% (last column in table 2(a)), which is closer to the 73.7%
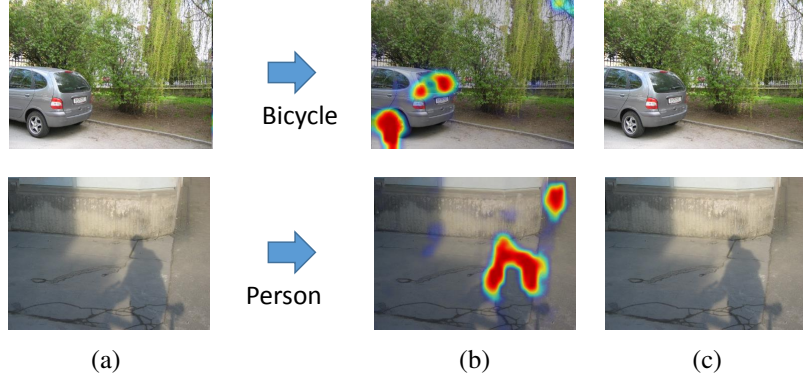
Figure 8: Removal of false detections on negative images by saliency refinement.(a) Input image, (b) false detections of bicycle model (row 1) and person model(row 2) before saliency refinement, (c) saliency-refined image with no false detections
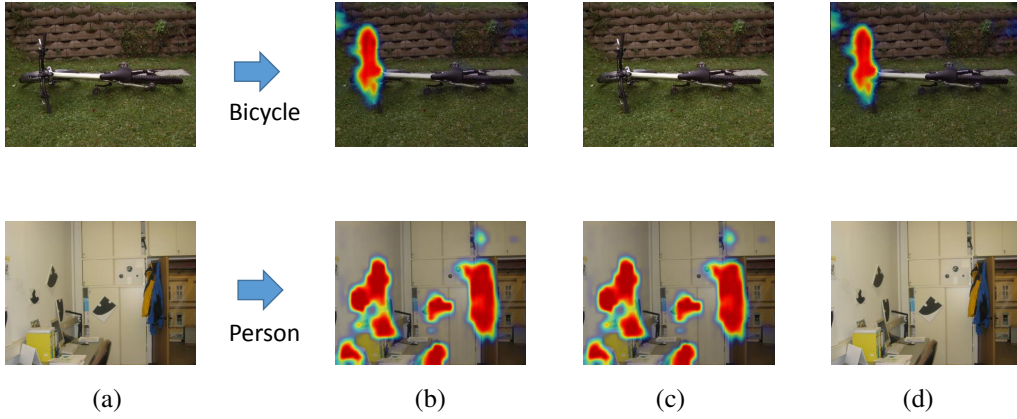

Figure 9: Failure cases of saliency refinement. (a) Input image, (b) true detections of bicycle model (top row) and false detections of person model (bottom row) before saliency refinement. The misclassifications of the classifier leads to errors in the final saliency map (c) with false negative in the top row and false positive in the bottom row for the bike and person models respectively. (d) An image classifier with 100% accuracy could avoid both of these errors

reported in [7] for 300 test images. The slight degradation in accuracy is due to the use of less number of training cycles in our model, to save training time. To achieve a faster training of the model, we used only 10 initial iterations of the saliency model training compared to 20 in [7]. Moreover, in these 10 iterations, we used only 150 training images in Train1 compared to 300 training images of [7]. With these two changes, our initial training time reduced to 30% of the time in [7] (10 iterations using 150 images vs 20 iterations using 300 images).

Effect of classifier feedback. In this experiment, we study the second stage of saliency modeling. The dictionaries obtained at the end of the first stage of saliency modeling are used to train the multi-class classifier using T2a training set and validated using T2b. Misclassified images from T2b are used to train the corresponding saliency model. Another iteration of this feedback

is carried out by interchanging T2a and T2b. There is an improvement of 2% in precision (column 4 of table 2(a)) due to classifier feedback, which is attributed to the car and person classes. The bike class is, by far, the easiest to model among the three as seen in [7] also, and hence the initial saliency models are able to capture the variability in most of the images. However, using the failed images to provide feedback from the classifier was not sufficient to improve the bike models. Compared to [7], there is an improvement of about 7% with only 10 iterations for saliency modeling and 2 iterations for classifier feedback. The results are supported qualitatively too, as shown by the saliency maps in Fig. 7.

Bicycle

Person

Car

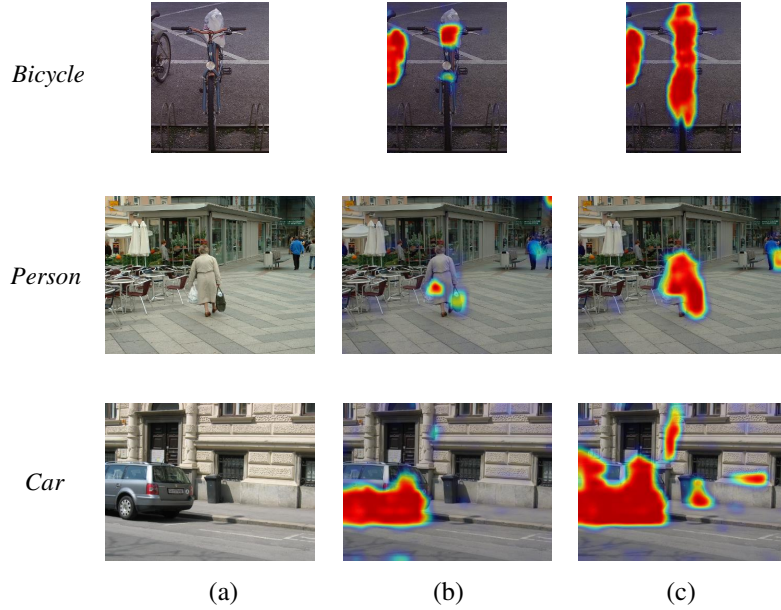(a)   (b)   (c)

Figure 10: Effectiveness of classifier feedback. (a) Input image, (b) saliency map of a model before classifier feedback and (c) saliency map of a model trained with classifier feedback. The *bicycle* and *person* images (c) show improvement in the saliency map due to feedback, while the *car* image (c) shows a failure case, where the classifier feedback introduced few false positives.

In Fig. 10, the saliency maps of *bicycle* and *person* images with distinct pose/features are improved by the classifier feedback, while on *car* image, even though the refinement improved the true positive detection of the saliency map, it also introduced some false detections along straight edges of the background clutter. Inspite of few exceptional cases, the classifier feedback helps to improve the saliency map especially on the images which has a rare pose or view point as shown in 10 (c) (*bicycle* , *person*). This is justified quantitatively too in Table 2.

Pixel-level result comparison. For comparison with a recent top-down saliency approach [20], we used their publicly available code and re-trained their models using T2 and evaluated on entire 600 test images. Pixel-level precision rates at EER (table 2(b)) show that the proposed patch-based approach achieves state-of-the-art result at pixel-level as well. Note that pixel-level saliency maps are generated from patch-level saliency maps as described in [7]. Despite using computationally intensive characteristics like 'objectness', and extracting color-based features from every superpixel, the performance of [20] drops when evaluated on the entire 600 test im-
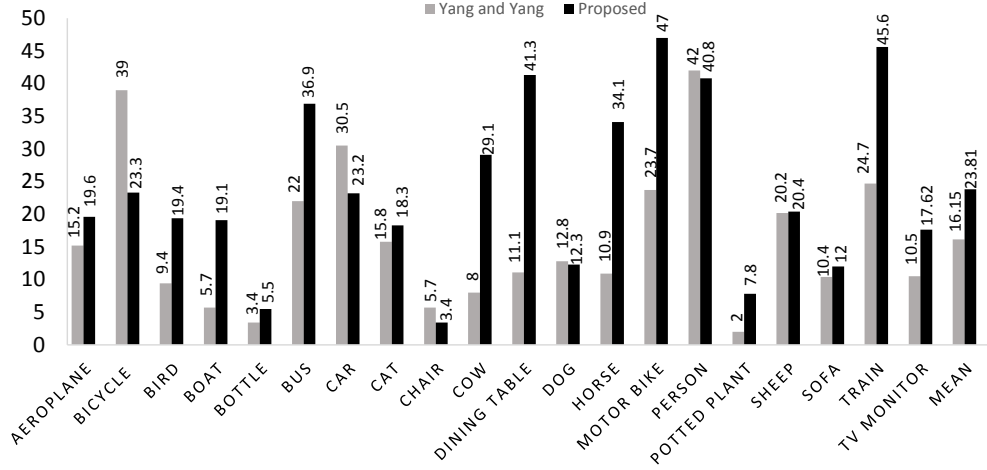
Figure 11: Patch-level precision rates (%) at EER on PASCAL VOC-07 compared to Yang and Yang [7].

ages, instead of evaluating on 150 same category and 150 background test images. This can be attributed to the inability to discriminate between object categories.

Comparison with other approaches. Our top-down saliency results are also compared with object localization/segmentation approaches that use precision rates at EER as evaluation metric. The proposed saliency models are evaluated on the test image set-up of shape mask [15], i.e, each model is evaluated only on 150 test images from their respective category. Table 3 shows the effectiveness of proposed saliency model by outperforming [16] and [15] in all the three categories. It is to be noted that [15] needs an additional level of supervision by manually labeling training images as *truncated* or *difficult*. Object class segmentation [31] extends [16] using superpixels as the basic unit for computation and the segmentation results are refined using a CRF operating on the superpixel graph. By maintaining identical parameters as our approach (without aggregating the histograms of a superpixel with its neighboring superpixels), [31] achieves a mean precision at EER of 54.56% which is lower than the proposed method operating on regular rectangular patches (61.4%).

7.2.2. PASCAL VOC 2007 dataset

We use all the available positive training (P1) and validation (P2) images to train the initial saliency models for 10 iterations. Since multiple classes are present in some images, we train one-vs-rest binary classifiers for each object using P1 images and validate on P2. Set of positive images with lower confidence are used to train respective saliency model incrementally.

Fig. 11 shows results for patch-level saliency estimation in which we achieve state-of-the-art performance in 15 out of 20 classes. Averaging over the entire 20 classes, we achieve a mean precision rate at EER (%) of 23.81 compared to 16.15 of [7], which is an improvement of 47%. We maintained the same dictionary size (512) and number of CRF weights as in [7]. The proposed saliency model trained for less number of cycles performs much better than [7] trained for 20 iterations. This is attributed to [7] failing in images having large dominant objects like bus or horse, as local patches of these contain limited relevant information. The additional classifier-guided dictionary training in the proposed model is able to capture this information leading to a
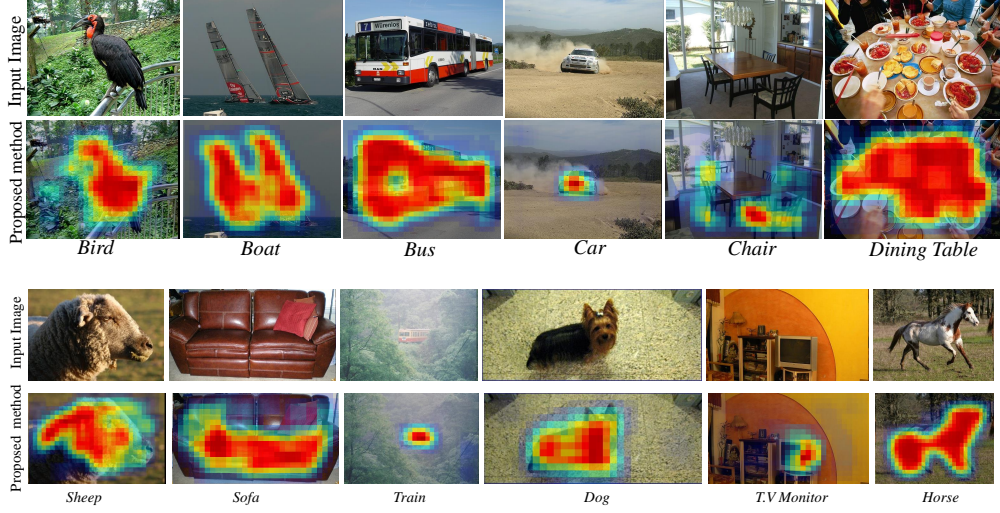
Figure 12: Saliency detection by the proposed method on PASCAL VOC-07.

significantly higher precison rate at EER for such classes.

Objects that are not visible clearly as in the *Train* image and those in the presence of clutter as in *TV Monitor* are correctly detected in Fig. 12. Even when a textureless object such as Sofa, is large and occupies almost the entire image, our method shows good accuracy in marking the area as salient.

Images with multiple objects

Fig. 13 shows the the ability of the proposed approach to discriminate among object categories on test images in which objects from multiple categories are simultaneously present. These qualitative results are supported by the improved performance in Fig. 11. Even when a boat is present in the *Person* image, the method marks only the person as salient. Similarly, bottles and bicycle are correctly detected even though person is dominating in the *bottle*, *Bicycle* images respectively. The presence of dominant TV monitor, makes cat detection a challenging task.Multiple instances of an object in *Cow* and *Motorbike* have been successfully marked as salient in Fig. 14.

Qualitative comparison

Fig. 15 compares the saliency results of Yang and Yang [7] and Kocak *et al.* [20] with our method. With the help of classifier-guided training, proposed method outperform [7, 20], especially on test images in which object size is too big compared to the patch size as can be observed from Fig. 15 (*Train*, *Cow*). In accordance with the precision rates at EER results, Dog and Person saliency maps of [7] are slightly better than proposed method (see *Dog* and *Person* in Fig. 15). Improved Performance of proposed method on less textured object classes like aeroplane, bottle and sofa is clearly visible in Fig. 15.
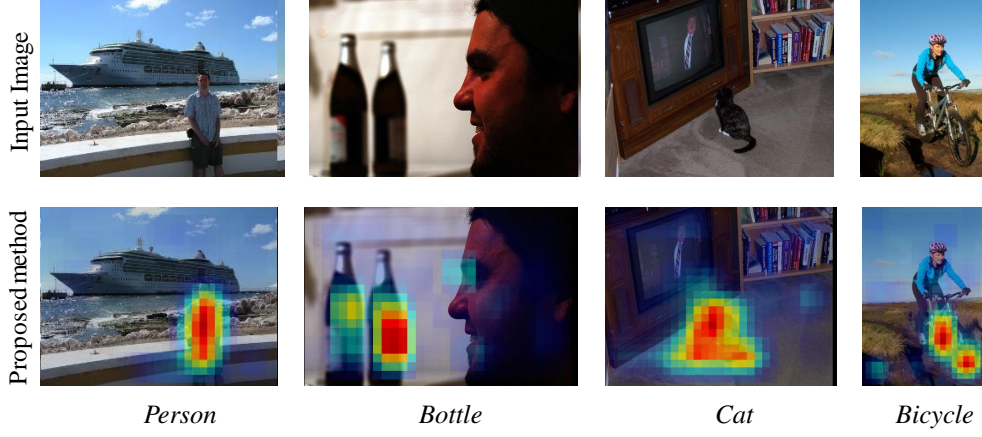
Figure 13: Images with multiple objects from different categories.

Figure 14: Images with multiple objects from same category.

7.3. Image segmentation

Although object segmentation in cluttered background is a challenging problem in itself, we investigate the effectiveness of using the saliency maps obtained by our method in segmentation. The saliency map generated is a probability map with 1 indicating the presence of object of interest and 0 indicating absence of object. Here non-object includes background as well as object pixels of negative classes. The saliency models of all categories are inferred on a test image and each pixel is assigned to the category of the saliency model that produces highest saliency value at that pixel. If the highest saliency at a pixel is below 0.5, those pixels are assigned to the background class. Fig. 16 shows segmentation results achieved by this simple thresholding . Even though our saliency models are trained and inferred at patch-level, it is capable of producing a segmentation that follows object boundaries. An improved pixel accurate segmentaion can be achieved by applying dedicated segmentation approaches such as Grabcut [40] on salient regions.

Graz-02 dataset

Intersection over union (IOU) is used as a metric to evaluate segmentation performance, computed as $IOU = TP/(TP + FP + FN)$, where TP is number of true positive pixels, FP is the number of false positive pixels and FN is number of false negative pixels. Table 4 compares the performance of the proposed method with [33] and [35] on the Graz02 dataset. Following [35], the proposed saliency models are evaluated on 150 test images from each object category ($3 \times$ 150).

Figure 15: Qualitative comparison with Yang and Yang [7] and Kocak *et al.*[20] on PASCAL VOC-07 dataset

<div align="center">Person   Bicycle   Car</div>
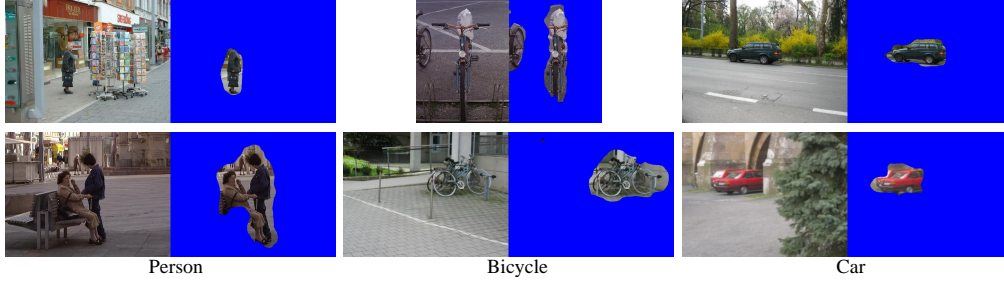
<div align="center">Figure 16: Segmentation from the saliency map by simple thresholding.</div>

Table 4: Comparison with state-of-the-art semantic segmentation tasks on 450 test images of Graz-02 dataset using intersection over union metric.

Algorithm	Singaraju and Vidal [33]	Jain *et al.* [35]	Proposed
Background	82.32	77.97	**83.46**
Bike	46.18	**55.6**	50.02
Car	36.49	**41.51**	40.81
Person	**38.99**	37.26	38.8
Mean	50.99	53.08	**53.27**

Even with a simple thresholding of the saliency map, the segmentations produced with our method outperforms both the methods. Better performance in background pixel classification illustrates the effect of classifier-weighted saliency inference, which reduced false positive detection.

PASCAL VOC-07 dataset

Image segmentation approach [32] evaluates their model using pixel classification accuracy; i.e percentage of pixels correctly classified for each category. To compare with this approach, the proposed model is also evaluated using this metric. The average classification accuracy (average accuracy of 21 categories- 20 object and one background category) obtained by thresholding saliency maps at 0.5 is better (29.66%) than image segmentation approach [32] (23%). Although our saliency maps are estimated using patch-level computations, the accuracy is comparable to that of the superpixel based segmentation approach of [31] (27% using 4 superpixel neighbors)

7.4. Image Classification

This paper proposes a novel and effective method for salient object detection in which the classifier plays an important role in improving results because of its ability to identify those objects whose models needs to be retrained by the CRF in order to remove false positives. Since there is a tight coupling between the saliency and classifier modules, it would be pertinent to investigate whether image classification performance gains with improved object models.

Graz-02 dataset

Table 5 compares the image classification accuracy of the classifier in the proposed framework with LLC [28], ScSPM [18] and [30]. In our implementation of ScSPM, we randomly

Table 5: Classification accuracy for Graz-02.

Algorithm	LLC [28]	Bilen *et al.* [30]	ScSPM [18]	ScSPM using proposed category-aware sparse coding	Proposed saliency-weighted classifier
Accuracy	87.8 ± 0.9	91.18 ± 1.4	88.5 ± 1	89.5 ± 1.2	**91.21 ± 1.2**

sample 100 features from each training image and form a dictionary of size 2048. As suggested in [28], 5 local dictionary atoms are used to code a feature in LLC. Although LLC is faster than ScSPM, its classification accuracy is less. When ScSPM is used in conjunction with our category-aware sparse coding, there is an improvement of about 1%. However, when the saliency information is incorporated in the form of a weight derived from the classifier, the classification accuracy is 91.21%, which is better than LLC and [30]. It may be noted that in [30], the results are obtained using SIFT features extracted from multiple patch sizes for training the classifier using the AUC criterion. In [41], the authors achieve an accuracy of 92.23% using multiple features such as color, shape and SIFT. Thus, although the classifier falls short of state-of-the-art performance by about 1%, the top-down saliency framework provides a reasonably good classifier as an accessory.

PASCAL VOC-07 dataset

In PASCAL VOC-07 dataset, objects from multiple categories are present in an image. So we train a binary SVM classifier for each object category as in [28] instead of one-vs-rest SVM used for Graz-02 dataset. For example, in an image that contains both cat and TV monitor, the cat binary classifier will estimate the presence of cat in the image and the TV monitor classifier will estimate the presence of TV monitor in the image. If both classifier respond positively, the image will be marked as one containing both cat and TV monitor. On the other hand if the cat classifier wrongly estimated the absence of cat, then it is considered as a false negative for cat category. Average precision of each category is evaluated seperately on all test images of PASCAL VOC-07 image classification dataset and the mean across 20 categories is evaluated. We achieved better mean average precision of 50.84% as compared to 50.65% by ScSPM and better classification in 12 out of 20 object classes. Although the improvement is not significant, most of the errors are due to inter-class similarity, e.g. between bike and motor bike classes. The joint saliency-classifier framework needs D_{sub} and a background dictionary in addition to the dictionaries for saliency modeling. These dictionaries are much smaller than the large dictionary (12,000 atoms) used by ScSPM.

7.5. Computation time

The training of joint framework for saliency estimation and image classification is faster as compared to the saliency estimation approaches of [20, 7], due to the reduced number of iterations and images used by proposed classifier-guided training. MATLAB implementations of all approaches were evaluated on a PC running on Intel Xeon 2.4GHz processor. Our training of all three object categories in Graz-02 and image classifier took just 3 hours and 34 minutes, while training of saliency models alone took 4 hours and 49 minutes by [7] and 30 hours and 10 minutes by [20].

The saliency estimation for 3 categories as well classifying the image took just 9.89 seconds. The SIFT feature extraction took 2.83 seconds, saliency inference per category took 1.72 seconds and category-aware sparse coding, saliency-weighted max-pooling, image classification and

saliency refinement took additional 1.9 seconds, totaling to 9.89 seconds ($3 \times 1.72 + 2.83 + 1.9 = 9.9$) seconds. The saliency estimation of 3 object categories alone took 7.99 seconds in [7]. The inference time is much larger in [20], which took 52 seconds. The ScSPM image classfication of [18] alone need 6.9 seconds, due to conventional sparse coding on D, while with the help of category-aware sparse coding, the additional computational time for classification reduces to 1.9 seconds only . For comparison, cascaded saliency and classification modules of [7] and [18] requires 14.89 seconds per image, while our joint approach gives better accuracy in 9.89 seconds.

8. Conclusion

In this work we propose a framework for top-down salient object detection. Since the pipeline of image classification [18] and top-down saliency [7] contains many common stages, our interconnected and mutually benefiting saliency-classification framework reduces the computational cost compared to their independent implementations. The image classifier is trained on novel category-aware sparse codes computed on object dictionaries used for saliency modeling. A novel saliency-weighted max-pooling is proposed to improve image classification by weighting the max-pooled vector in each block of the spatial pyramid with a weight derived from net saliency of that block. Similarly, saliency maps are improved by using the image classifier that leverages information about presence of the object in an image. In the current implementation we extracted SIFT features using a fixed patch size, that reduces the performance of the model, if the object size is too small compared to the patch size. In the future, we will implement a multiple-patch size, multi-feature based joint framework for saliency estimation and image classification.

9. References

References

[1] J. Li, W. Gao, Visual Saliency Computation-A Machine Learning Perspective, Lecture Notes in Computer Science, Springer, 2014.

[2] J. Li, Y. Tian, T. Huang, Visual saliency with statistical priors, International journal of computer vision 107 (3) (2014) 239–253.

[3] Y. Jia, M. Han, Category-independent object-level saliency detection, in: ICCV, 2013.

[4] L. Itti, C. Koch, Feature combination strategies for saliency-based visual attention systems, Journal of Electronic Imaging 10 (1) (2001) 161–169.

[5] Q. Zhao, C. Koch, Learning a saliency map using fixated locations in natural scenes, Journal of vision 11 (3) (2011) 9–9.

[6] V. Navalpakkam, L. Itti, Search goal tunes visual features optimally, Neuron 53 (4) (2007) 605–617.

[7] J. Yang, M.-H. Yang, Top-down visual saliency via joint crf and dictionary learning, in: CVPR, 2012.

[8] D. Gao, S. Han, N. Vasconcelos, Discriminant saliency, the detection of suspicious coincidences, and applications to visual recognition, IEEE Transactions on Pattern Analysis and Machine Intelligence 31 (6) (2009) 989–1005.

[9] A. Torralba, A. Oliva, M. S. Castelhano, J. M. Henderson, Contextual guidance of eye movements and attention in real-world scenes: the role of global features in object search., Psychological Review 113.

[10] G. Sharma, F. Jurie, C. Schmid, Discriminative spatial saliency for image classification, in: CVPR, 2012.

[11] B. Yao, A. Khosla, L. Fei-Fei, Combining randomization and discrimination for fine-grained image categorization, in: CVPR, 2011.

[12] R. Girshick, J. Donahue, T. Darrell, J. Malik, Rich feature hierarchies for accurate object detection and semantic segmentation, in: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2014.

[13] B. Alexe, T. Deselaers, V. Ferrari, What is an object?, in: CVPR, 2010.

[14] R. G. Cinbis, J. Verbeek, C. Schmid, et al., Multi-fold mil training for weakly supervised object localization, in: CVPR 2014-IEEE Conference on Computer Vision & Pattern Recognition, 2014.

[15] M. Marszałek, C. Schmid, Accurate object recognition with shape masks, International journal of computer vision 97 (2) (2012) 191–209.

[16] B. Fulkerson, A. Vedaldi, S. Soatto, Localizing objects with smart dictionaries, in: ECCV, 2008.

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

[18] J. Yang, K. Yu, Y. Gong, T. Huang, Linear spatial pyramid matching using sparse coding for image classification, in: CVPR, 2009.

[19] F. Shahbaz Khan, J. van de Weijer, M. Vanrell, Top-down color attention for object recognition, in: ICCV, 2009.

[20] A. Kocak, K. Cizmeciler, A. Erdem, E. Erdem, Top down saliency estimation via superpixel-based discriminative dictionaries, in: BMVC, 2014.

[21] K. Chatfield, V. Lempitsky, A. Vedaldi, A. Zisserman, The devil is in the details: an evaluation of recent feature encoding methods, in: BMVC, 2011.

[22] R.-E. Fan, K.-W. Chang, C.-J. Hsieh, X.-R. Wang, C.-J. Lin, Liblinear: A library for large linear classification, The Journal of Machine Learning Research 9 (2008) 1871–1874.

[23] J. Zhu, Y. Qiu, R. Zhang, J. Huang, W. Zhang, Top-down saliency detection via contextual pooling, Journal of Signal Processing Systems 74 (1) (2014) 33–46.

[24] N. Khan, M. F. Tappen, Discriminative dictionary learning with spatial priors., in: ICIP, 2013, pp. 166–170.

[25] F. Khan, J. Weijer, M. Vanrell, Modulating shape features by color attention for object recognition, IJCV 98 (1) (2012) 49–64.

[26] F. Moosmann, D. Larlus, F. Jurie, Learning saliency maps for object categorization, in: International Workshop on the Representation and Use of Prior Knowledge in Vision (ECCV workshop), Springer, Springer, 2006.
URL http://lear.inrialpes.fr/pubs/2006/MLJ06

[27] S. Lazebnik, C. Schmid, J. Ponce, Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories, in: CVPR, 2006.

[28] J. Wang, J. Yang, K. Yu, F. Lv, T. Huang, Y. Gong, Locality-constrained linear coding for image classification, in: CVPR, 2010.

[29] T. Ge, K. He, J. Sun, Product sparse coding, in: CVPR, 2014.

[30] H. Bilen, V. Namboodiri, L. Gool, Object and action classification with latent window parameters, IJCV 106 (3) (2014) 237–251.

[31] B. Fulkerson, A. Vedaldi, S. Soatto, Class segmentation and object localization with superpixel neighborhoods, in: CVPR, 2009.

[32] D. Aldavert, A. Ramisa, R. L. de Mantaras, R. Toledo, Fast and robust object segmentation with the integral linear classifier, in: CVPR, 2010.

[33] D. Singaraju, R. Vidal, Using global bag of features models in random fields for joint categorization and segmentation of objects, in: CVPR, IEEE, 2011, pp. 2313–2319.

[34] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, A. Zisserman, The PASCAL Visual Object Classes Challenge 2007 (VOC2007) Results, http://www.pascal-network.org/challenges/VOC/voc2007/workshop/index.html.

[35] A. Jain, L. Zappella, P. McClure, R. Vidal, Visual dictionary learning for joint object categorization and segmentation, in: ECCV, 2012.

[36] T. Serre, L. Wolf, T. Poggio, Object recognition with features inspired by visual cortex, in: CVPR, 2005.

[37] H. Lee, A. Battle, R. Raina, A. Y. Ng, Efficient sparse coding algorithms, in: NIPS, 2007.

[38] A. Opelt, A. Pinz, M. Fussenegger, P. Auer, Generic object recognition with boosting, PAMI 28 (3) (2006) 416–431.

[39] K. Crammer, Y. Singer, On the algorithmic implementation of multiclass kernel-based vector machines, The Journal of Machine Learning Research 2 (2002) 265–292.

[40] C. Rother, V. Kolmogorov, A. Blake, Grabcut: Interactive foreground extraction using iterated graph cuts, in: ACM Transactions on Graphics (TOG), Vol. 23, ACM, 2004, pp. 309–314.

[41] Y.-H. Tsai, J. Yang, M.-H. Yang, Decomposed learning for joint object segmentation and categorization, in: BMVC, 2013.