Region Ranking SVM for Image Classification

Zijun Wei Minh Hoai
Stony Brook University, Stony Brook, NY 11794, USA
{zijwei,minhhoai}@cs.stonybrook.edu

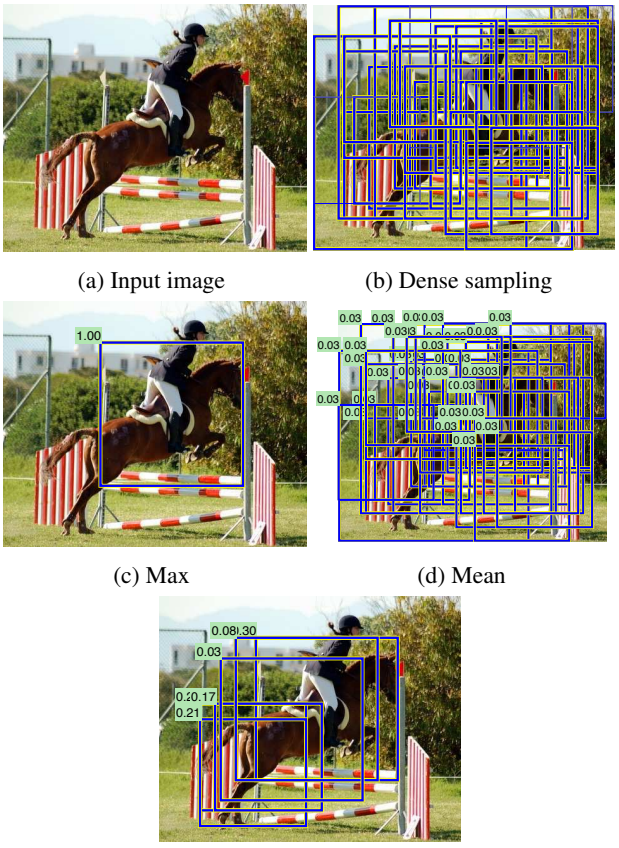
Abstract

The success of an image classification algorithm largely depends on how it incorporates local information in the global decision. Popular approaches such as average-pooling and max-pooling are suboptimal in many situations. In this paper we propose Region Ranking SVM (RRSVM), a novel method for pooling local information from multiple regions. RRSVM exploits the correlation of local regions in an image, and it jointly learns a region evaluation function and a scheme for integrating multiple regions. Experiments on PASCAL VOC 2007, VOC 2012, and ILSVRC2014 datasets show that RRSVM outperforms the methods that use the same feature type and extract features from the same set of local regions. RRSVM achieves similar to or better than the state-of-the-art performance on all datasets.

1. Introduction

Image classification is one of the most fundamental and challenging tasks in computer vision. Image classification aims at recognizing the semantic category of an image, such as whether the image contains a certain object (e.g., bicycle, car), depicts a certain scene (e.g., beach, bedroom), or captures a certain action (e.g., answering phone, riding horse). Recognizing the semantic category of an image, however, is challenging because the location of the *semantic region*, the image area that corresponds to the semantic category that we need to recognize, is unknown.

A popular approach is to tackle this problem is to aggregate the information computed at multiple regions of an image. Recent examples of this approach are to average CNN feature vectors [4, 18, 31] that are computed at multiple locations and scales [30, 35]. These approaches lead to impressive recognition performance on a number of datasets, including PASCAL VOC [7] an ImageNet [27]. However, average pooling does not always yield good performance, especially when the semantic region is small and has little or no overlap with the majority of the local regions being considered. In this case, average pooling harms recognition due to too much background noise.



(a) Input image (b) Dense sampling

(c) Max (d) Mean

(e) RRSVM

Figure 1: **Image classification with unknown object location.** A popular approach is to consider multiple local regions (b), and uses either max-pooling (c) or average-pooling (d). We propose a method that selectively uses the regions and combine them for classification (e).

Another popular approach is to treat the semantic region as a latent variable and use a *region classifier* to localize the semantic region. A region classifier is used to evaluate multiple regions of an image, and the one that yields the maximum classifier score is considered the semantic region. This approach assumes the semantic region is among the local regions being considered, or significantly overlaps

with one of them. This assumption, however, does not usually hold in practice because: (i) the local regions may be required to have some certain shape (e.g., axis parallel rectangular regions, while the semantic region can be free-form and dis-contiguous), and (ii) the total number of regions that can be considered must be limited to meet a computational budget. For example, CNN features [4, 18, 31] can only be computed on square regions of a certain size. Also, computing CNN features is time consuming so the total number of regions is usually limited to ten [18] or at most several hundred [31, 35]. Furthermore, this approach might not work well even when the semantic region is among the regions being considered: there is no guarantee that the region that yields the maximum score would actually correspond to the semantic region. This is especially true when the region classifier is far from perfect; it is often trained without region-level labels because training images are generally not divided into annotated regions.

Several prior works also suggest the inefficacy of using the maximum score for classification [6, 13, 16]. Hu et al. [16] considered both the maximum and the average scores, and reported better classification performance for the maximum score in many experiments. Hoai and Zisserman [13] considered the problem of recognizing human actions in video where the exact locations of human actions are unknown. To tackle temporal uncertainty, they posed it as a multiple instance learning problem where subsequences of a video clip were considered as candidates for the actual location of a human action. Assuming there was a base classifier, they observed the inadequacy of using the maximum score of the subsequences as the decision value. Instead, they proposed Subsequence Score Distribution (SSD), a decision function that was based on the distribution of the scores of the video subsequences. They showed that SSD outperformed the method that used the maximum score by a wide margin. Although SSD was more robust than the maximum score, its performance largely depends on the performance of the base classifier, which had been learned independently of SSD.

In this paper, we propose Region Ranking SVM (RRSVM), a novel formulation for the classification of multiple regions. RRSVM is based on SSD, which uses all region scores instead of using just the max (e.g., [15, 23]) or the mean for classification. RRSVM trains an SVM base classifier and uses the scores of *all* the regions in an image to predict the image label. The scores of the regions in an image are ranked and combined using a weight vector, called region-combination vector. This vector is common to all images of a category, and it is *jointly* learned with the base classifier. The region-combination vector can be considered as a classifier of which the input is the distribution of region scores. The distribution preserves more information than both extreme (i.e., max) and summary (i.e., mean) statistics, and it will be empirically shown to be more robust and effective. Figure 1 illustrates the essence of our approach.

RRSVM bares some similarities to bilinear models [26, 36]. Bilinear models also consider the correlation between multiple data instances and learn two weight vectors for factorizing 'styles' and 'contents'. However, bilinear models require a fixed and known ordering of the instances, which is not applicable to image classification problems. In contrast, RRSVM automatically determines the order of regions by ranking them using the scores obtained from the base classifier.

2. Related Work

2.1. Image Classification

Image classification is one of the most important problems in computer vision. Training SVM classifiers on Fisher Vectors (FVs) [25, 28] and variant methods [1, 3, 5, 17, 22, 44] have once been the dominating methods for large scale image classification problems. Recent development in deep learning and Convolution Neural Networks [18, 21, 30, 31, 35, 43] have significantly advanced the performance of large scale image classification. Many algorithms, either based on Fisher Vectors or CNN, often compute an average feature vector over multiple local regions. This often oversimplifies the situations that actually the different local regions of an image contain different amount of information for image classification. Unlike existing works, our proposed method selectively integrates information into the classification decision.

Many works have been proposed [2, 11, 20, 41] to utilize local information for classification tasks. Among these the most widely used is the spatial pyramid representation [20]. Spatial pyramid representation, however, relies on rigid geometric correspondence of grid division, which ignores the importance of semantic or discriminative localization. This model has limited discriminative power for recognizing semantic category with huge variance in location.

2.2. Subsequence Score Distribution (SSD)

SSD [13] is a method for considering the score distribution of multiple feature vectors. SSD was initially proposed for human action recognition. The technique is general and can be applied to image classification. For convenience, here we review it in the context of image classification.

SSD assumes a base classifier f for evaluating image regions has been learned. Consider an image region $\mathbf{x}$, ideally, $f(\mathbf{x})$ should be 1 if $\mathbf{x}$ is a region that contains the target object and -1 otherwise. Given a test image $\mathbf{B}$, instead of using the maximum score $\max_{\mathbf{x}\in\mathbf{B}} f(\mathbf{x})$ as the decision value for $\mathbf{B}$, SSD proposes to learn an aggregation operator as below.

Consider the training data that consists of n images

$\{\mathbf{B}_i\}_{i=1}^n$ and associated image labels $\{y_i\}_{i=1}^n$. Assume for now that all images have the same number of sampled regions. Let m be the number of regions and d the dimension of each region descriptor. SSD represents each image as a matrix $\mathbf{B}_i \in \Re^{d\times m}$, but the order of the columns can be arbitrary. Let $\mathbf{b}_i^1, \cdots, \mathbf{b}_i^m$ be the region descriptors in image $\mathbf{B}_i$. Let $\mathbf{a}_i = [sort(f(\mathbf{b}_i^1), \cdots, f(\mathbf{b}_i^m))]^T$. Here, *sort* is the function that reorders the inputs in descending order. $\mathbf{a}_i$ represents the score distribution of regions from $\mathbf{B}_i$. SSD learns score-combination vector $\mathbf{s}$ by optimizing the following objective:

$$\underset{\mathbf{s},b}{\text{minimize}} \sum_{i=1}^{n} \max(1 - y_i(\mathbf{s}^T\mathbf{a}_i + b), 0) \quad (1)$$

$$\text{s.t. } s_1 \geq s_2 \geq \cdots \geq s_m \geq 0, \quad (2)$$

$$\sum_{j=1}^{m} s_j = 1. \quad (3)$$

The above optimization problem seeks a weight vector $\mathbf{s}$ and the bias term b for separating between the score distribution vectors of positive and negative data. The objective in Eq. (1) is the sum of Hinge losses, as used in the objective of SVMs [37].

SSD has been shown to yield impressive results on human action recognition [13]. However, the performance of SSD depends heavily on the base classifier. In [13], the base classifier is simply trained to separate the mean vectors of positive videos and the mean vectors of negative videos. This base classifier may not work well for other problems. Unfortunately, a straight forward formulation for iterative learning of the base classifier and the SSD classifier will lead to a degenerate solution. This will be explained in Sec. 3.2.

3. Region-Ranking SVM (RRSVM)

3.1. RRSVM Formulation

RRSVM is a framework that jointly learns the base classifier and the SSD decision function. Using the same notation as in Sec. 2.2, RRSVM is posed as the following optimization problem:

$$\underset{\mathbf{w},\mathbf{s},b}{\text{minimize}} \ \lambda||\mathbf{w}||^2 + \sum_{i=1}^{n}(\mathbf{w}^T\Gamma(\mathbf{B}_i;\mathbf{w})\mathbf{s} + b - y_i)^2 \quad (4)$$

$$\text{s.t. } s_1 \geq s_2 \geq \cdots \geq s_m \geq 0, \quad (5)$$

$$h(\{\Gamma(\mathbf{B}_i;\mathbf{w})\mathbf{s}\}) \leq 1. \quad (6)$$

In the above formulation, $\mathbf{w}$ and b are the weight vector and the bias term of the SVM base classifier. $\Gamma(\mathbf{B};\mathbf{w})$ denotes a matrix that can be obtained by rearranging the columns of the matrix $\mathbf{B}$ so that $\mathbf{w}^T\Gamma(\mathbf{B};\mathbf{w})$ is a sequence of non-increasing values. This sequence is the sorted list of the SVM scores of individual regions, which will be referred to as the *region-score vector*. This region-score vector is essentially a non-parametric representation for the score distribution of the regions in an image $\mathbf{B}$. Vector $\mathbf{s}$ is the weight vector for combining the SVM region scores for each image; this vector is common to all images of a class.

The objective of the above formulation consists of the regularization term $\lambda||\mathbf{w}||^2$ and the sum of regression losses. There, λ is the only parameter of RRSVM and it is tunable. This objective is analogous to the objective of Least-Squares SVMs (LSSVM) [33]. LSSVM, also known as kernel Ridge regression [29]. LSSVM has been shown to perform equally well as SVM in many classification benchmarks [34, 39]. We base our formulation on LSSVM instead of SVM [37] because LSSVM has a closed-form solution, which is a computational advantage over SVM.

Constraint (5) requires $\mathbf{s}$ to be non-negative and monotonic. This is to emphasize the relative importance of higher region-classification scores (recall that we take the dot product between $\mathbf{s}$ and $\mathbf{w}^T\Gamma(\mathbf{B};\mathbf{w})$, the sorted vector of local region scores).

Constraint (6) requires the feature vectors to be bounded. Here $h(\cdot)$ is the function that measures the spread of a set of vectors:

$$h(\{\mathbf{x}_i\}_{i=1}^n) = \sum_{i=1}^{n} \left|\left| \mathbf{x}_i - \frac{1}{n}\sum_{i=1}^{n}\mathbf{x}_i \right|\right|^2 \quad (7)$$

Recall that $\Gamma(\mathbf{B}_i;\mathbf{w})\mathbf{s}$ is a linear combination of local region feature vectors of $\mathbf{B}_i$; it is essentially the representation vector for the image $\mathbf{B}_i$. Constraint (6) requires the spread of these representation vectors to be bounded by 1. In theory, this constraint governs the compactness of the feature vectors, which is important for the theoretical analysis of the max-margin learning framework [37]. In practice, this constraint prevents $\mathbf{s}$ to become extremely large. To see why this is important, consider the loss terms in Eq. (4). These loss terms remain the same if $\mathbf{w}$ and $\mathbf{s}$ are scaled inversely. Thus, if $\mathbf{s}$ can be made extremely large then $\mathbf{w}$ can be made extremely small; rendering the margin term $\lambda||\mathbf{w}||^2$ ineffective.

The right hand side of Constraint (6) is rather arbitrary. We can replace 1 by any positive value γ, and obtain an equivalent formulation by adjusting the value of λ as follows: $\lambda_{new} := \gamma\lambda_{old}$.

Once $(\mathbf{w}, b, \mathbf{s})$ have been learned, the classification decision value for a test image $\mathbf{B}$ is taken as:

$$\mathbf{w}^T\Gamma(\mathbf{B};\mathbf{w})\mathbf{s} + b. \quad (8)$$

As can be seen, this decision function is the same as the decision function of SSD, which has been shown to be very effective [13].

3.2. Alternative Formulations

Compare SSD (Eq. 1) and RRSVM (Eqs. 4 & 11), one can notice the two differences in the formulations for learning $\mathbf{s}$. First, SSD uses Hinge loss while RRSVM uses squared loss. This modification is necessary to ensure that $\mathbf{w}$ and $\mathbf{s}$ optimizes the same objective.

Second, the L_1-norm constraint for $\mathbf{s}$ in Eq. (3) is replaced by the bound on the spread of representation vectors in Eq. (6). Apart from the theoretical reasons for using Eq. (6) as explained above, there are potential issues for using the L_1-norm constraint.

Consider an alternative RRSVM formulation where Eq. (6) is replaced by Eq. (3). This formulation often leads to a solution where $\mathbf{s}$ degenerates to the solution $s_1 = 1$ and $s_2 = \cdots = s_m = 0$. This has been observed in many cases of our experiments. This tendency can be explained as follows. In the learning formulation, the margin term $\lambda||\mathbf{w}||^2$ is included in the objective. Thus, the formulation prefers small $||\mathbf{w}||$. Consider the loss terms in the objective, these terms remain the same if $||\mathbf{w}||$ and $||\Gamma(\mathbf{B}_i;\mathbf{w})\mathbf{s}||$ are scaled inversely. Thus, in general, the bigger $\{||\Gamma(\mathbf{B}_i;\mathbf{w})\mathbf{s}||\}$ are, the smaller $||\mathbf{w}||$ can become. But, for any image $\mathbf{B}$, the vector $\mathbf{s}$ that leads to highest value of $||\Gamma(\mathbf{B};\mathbf{w})\mathbf{s}||$ is the degenerated solution given above. To see this, suppose $\Gamma(\mathbf{B};\mathbf{w}) = [\mathbf{b}^1, \cdots, \mathbf{b}^m]$ and the local feature vectors of $\mathbf{B}$ are L_2-normalized (i.e., $||\mathbf{b}^1|| = \cdots = ||\mathbf{b}^m|| = 1$). Using the Cauchy-Schwarz inequality, we have:

$$||\Gamma(\mathbf{B};\mathbf{w})\mathbf{s}||^2 = ||\sum_{i=1}^{m} s_i \mathbf{b}^i||^2 \tag{9}$$

$$\leq (\sum_{i=1}^{m} s_i)(\sum_{i=1}^{m} s_i ||\mathbf{b}^i||^2) = 1 \tag{10}$$

In general, $\mathbf{b}^i \neq \mathbf{b}^j$, so equality holds only when $s_1 = 1$ and $s_2 = \cdots = s_m = 0$. Thus, the alternative learning formulation is biased toward the degenerated solution. The formulation for joint learning of $\mathbf{w}$ and $\mathbf{s}$ should avoid using the L_1-norm constraint. Note that the usage of this constraint in the SSD formulation is fine because $\mathbf{w}$ is fixed.

3.3. Optimization

The learning formulation given in Eq. 4 can be optimized with block coordinate descent, alternating between the following two procedures:

(A) Fix $\mathbf{w}$, optimize Eq. (4) w.r.t. $\mathbf{s}$ and b,

(B) Fix $\mathbf{s}$, optimize Eq. (4) w.r.t. $\mathbf{w}$ and b.

For Procedure (A), $\mathbf{w}$ is fixed, and let $\widehat{\mathbf{B}}_i$ be $\Gamma(\mathbf{B}_i;\mathbf{w})$ and $\mathbf{a}_i$ be $\widehat{\mathbf{B}}_i^T\mathbf{w}$. Procedure (A) is equivalent to:

$$\underset{\mathbf{s},b}{\text{minimize}} \sum_{i=1}^{n} (\mathbf{a}_i^T\mathbf{s} + b - y_i)^2 \tag{11}$$

$$\text{s.t. } s_1 \geq s_2 \geq \cdots \geq s_m \geq 0, \tag{12}$$

$$\mathbf{s}^T \left(\sum_{i=1}^{n} \widehat{\mathbf{B}}_i^T \widehat{\mathbf{B}}_i - \frac{1}{n} (\sum_{i=1}^{n} \widehat{\mathbf{B}}_i)^T (\sum_{i=1}^{n} \widehat{\mathbf{B}}_i) \right) \mathbf{s} \leq 1. \tag{13}$$

The above is a Quadratically Constrained Quadratic Program (QCQP). This optimization problem is convex, and it can be solved efficiently and globally using a QCQP solver, such as IBM Cplex[1].

For Procedure (B), $\mathbf{s}$ is fixed, and let $(\mathbf{w}_{old}, b_{old})$ are the current values of $(\mathbf{w}, b)$. Procedure (B) seeks new values for $(\mathbf{w}, b)$ to decrease the objective of Eq. (4). Let $\mathbf{u}_i$ be $\Gamma(\mathbf{B}_i;\mathbf{w}_{old})\mathbf{s}$, we first solve the following ridge regression problem to get $(\mathbf{w}^*, b^*)$:

$$\mathbf{w}^*, b^* := \underset{\mathbf{w},b}{\text{argmin}} \lambda||\mathbf{w}||^2 + \sum_{i=1}^{n} (\mathbf{w}^T\mathbf{u}_i + b - y_i)^2 \tag{14}$$

There exists a closed form solution for the above optimization problem, and therefore $(\mathbf{w}^*, b^*)$ can be found efficiently. $(\mathbf{w}^*, b^*)$ define the line search direction for new values of $(\mathbf{w}, b)$; we perform binary search until we find a feasible solution with lower objective value as follows. Let $g(\mathbf{w}, b, \mathbf{s})$ denote the objective value of Eq. (4). If $g(\mathbf{w}^*, b^*, \mathbf{s}) \leq g(\mathbf{w}_{old}, b_{old}, \mathbf{s})$ and $h(\{\Gamma(\mathbf{B}_i;\mathbf{w}^*)\}) \leq 1$ then we terminate the Procedure (B) and output $\mathbf{w}_{new} := \mathbf{w}^*, b_{new} := b^*$. Otherwise, let $\mathbf{w}^* := (\mathbf{w}^* + \mathbf{w}_{old})/2, b^* := (b^* + b_{old})/2$ and we check the objective value an the constraint satisfaction again. If the number of binary splits exceeds 10, we terminate the procedure and return $\mathbf{w}_{old}, b_{old}$. In practice, this procedure usually requires no binary split and always terminate within six splits.

The block-coordinate descent algorithm is guaranteed to converge because each procedure does not increase the objective value. In our experiments, it usually converges within 15 iterations.

We initialize the algorithm by: i) set $s_1 = \cdots s_m = 1/m$, ii) represent each image by the mean of its local region features, and iii) train a binary SVM to get initial $\mathbf{w}$ and b. This initialization and the optimization algorithm discussed in this section work well in our experiments, but smarter initialization strategies or better optimization policies can be used, e.g., [9, 19, 32].

As discussed in Sec. 3.1, the right hand side of Constraint (6) is rather arbitrary. In practice, it is convenient to set it to be the spread of the initial image representation vectors (the means of images' local region descriptors in our experiments). Learning $\mathbf{s}$ can be considered as learning

[1] www-01.ibm.com/software/commerce/optimization/cplex-optimizer/

the new feature representation, and this constraint requires the space contraction of the new feature representation.

3.4. Extensions

One way of extending the formulation presented in Sec. 3.1 is to replace Γ by another appropriate function. This function must take an unordered set of vectors and return a matrix of fixed dimensions. These dimensions must be matched by the dimensions of $\mathbf{w}$ and $\mathbf{s}$. For example:

- If the number of regions of each image (m) is high, Γ can be replaced by a function that returns the m' highest scored regions in descending order and set the dimension of $\mathbf{s}$ to m', with $m' \ll m$.
- If it is impossible to enumerate and find regions with the highest scores, Γ can be combined with random sampling, which represents a distribution by its random samples.
- If the number of the regions from each image differ, Γ can be replaced by a function that combines sorting and region selection. This selection procedure could simply use random sampling with replacement, or it could deterministically cycle through the regions in the decreasing order of the region scores.

4. Experiments

We perform experiments on three classification benchmarks: image classification on the PASCAL VOC 2007 [7] dataset, human action recognition in still images on the PASCAL VOC 2012 [8] dataset, and the object classification challenge of the ImageNet Large Scale Visual Recognition Challenge 2014 (ILSVRC 2014) [27]. In our experiments, we compare RRSVM with several competing methods that use the same feature representation. While RRSVM is agnostic to the particular feature representations, we use features extracted from the pre-trained VGG-D network [31], which has 16 layers and is commonly referred to as VGG16. We use VGG16 because of its excellent performance, being the best among methods that use a single neural network. Our feature extraction is based on the MatConvNet library [38].

4.1. Feature Extraction

Our feature extraction pipeline is the same as described in [31]. The last fully connected layer of VGG16 is removed and the remaining fully connected layer is converted to a fully convolutional layer, similar to [30]. To compute feature vectors for an image, the image is first resized isotropically so that the smallest size is equal to a predefined size Q, where $Q \in \{256, 384, 512\}$. Additionally, the resized images are flipped horizontally, creating six images at three scales. The fully-convolutional net is applied to each of the six images, leading to a feature map with 4096 channels. The size of feature map depends on the size of the resized image. Combining the feature vectors at three scales and with and without flipping, each input image typically yields 200 to 400 densely sampled feature vectors, each feature vector can be mapped back to a subwindow of the original input image.

Our work differs from [31] only in the feature pooling step. In [31], average pooling is applied on each image scale. Meanwhile, we compute the averaged of all feature vectors of all scales and subsequently perform L_2-normalization to create a global feature descriptor. We also perform L_2 normalization on each densely sampled feature vectors to create local feature descriptors. Finally, we stack the global feature descriptor with each local feature descriptor to get a 8192-dim vector to represent a local region of the input image. We do not use multi-crop sampling [35], which is very computationally demanding.

4.2. PASCAL VOC 2007 Object Classification

We first perform experiments on the PASCAL VOC 2007 dataset [7]. This dataset consists of three disjoint subsets for training, validation and testing; and they contain 2501, 2510 and 4952 images respectively. Each image is annotated with one or several labels, corresponding to 20 object categories. Classification performance for each category is measured by Average Precision. The mean Average Precision of all categories is denoted as mAP.

The baseline of our method is the one that trains a classifier on the average of all feature vectors computed for an image, as in [31]. We consider this as the fairest method to compare with RRSVM because they use the same set of feature vectors. In this paper, we use LSSVM as the classifier because of its computational efficiency and competitive performance [12, 34, 39, 42]. The top two rows in Table 1 show that LSSVM performs similar to SVM (with the best C tuned using 5-fold cross validation). Hereafter, we use LSSVMs as baselines for all the remaining experiments. We used the validation set for tuning parameter λ of LSSVM and RRSVM. We found that the best performance is achieved when $\lambda = 10^{-4}n$, where n is the total number of training data. For final classification results, we trained classifiers on the combined training and validation data and the average precision is reported in Table 1.

There are several noticeable results in Table 1. First, for the baseline method (denoted as LSSVM), the mAP is slightly higher than 89.3% of VGG16 reported in [31]. This may be due to the slight difference in the feature normalization step. Second, LSSVM-SSD and LSSVM-Max perform similarly. This might be because SSD elects to use a single effective region in this situation. Third, there is a significant performance gap between RRSVM and the baseline LSSVM, and RRSVM also outperforms all other meth-

Model	aero	bike	bird	boat	bottle	bus	car	cat	chair	cow	table	dog	horse	mbike	prsn	plant	sheep	sofa	train	tv	mAP
SVM	98.6	95.9	97.1	95.8	71.2	91.0	93.6	96.1	72.9	86.6	88.3	96.3	96.7	94.0	97.4	70.5	92.8	82.5	97.4	89.9	90.2
SVM-Max	98.6	97.3	97.4	96.6	78.3	93.0	95.2	96.9	76.3	87.2	88.4	96.8	97.0	95.0	98.5	75.1	93.2	83.7	97.7	92.4	91.7
LSSVM	98.9	95.6	97.2	95.4	69.5	91.1	93.5	96..4	73.9	87.4	88.7	96.5	96.9	94.1	97.1	70.3	93.7	83.5	98.3	88.4	90.3
LSSVM-Max	98.9	96.9	97.9	95.7	75.1	92.8	94.7	97.1	76.9	88.7	**88.9**	97.2	**97.1**	95.2	98.2	75.5	95.0	84.3	98.4	91.4	91.8
LSSVM-SSD	98.9	96.8	97.7	95.8	75.0	92.7	94.4	97.0	76.9	89.0	**88.9**	97.3	**97.1**	95.2	98.2	75.5	95.0	84.3	98.4	91.3	91.8
RRSVM	**99.2**	**97.4**	**98.1**	**96.7**	**79.7**	**94.5**	**95.9**	**97.4**	**79.3**	**89.3**	**88.9**	**97.7**	**97.1**	**95.7**	**98.8**	**79.5**	**95.4**	**84.8**	**98.6**	**93.1**	**92.9**
Chatfield [4]	95.3	90.4	92.5	89.6	54.4	81.9	91.5	91.9	64.1	76.3	74.9	89.7	92.2	86.9	95.2	60.7	82.9	68.0	95.5	74.4	82.4
Wei et al. [40]	96.0	92.1	93.7	93.4	58.7	84.0	93.4	92.0	62.8	89.1	76.3	91.4	95.0	87.8	93.1	69.9	90.3	68.0	96.8	80.6	85.2
VGG16 [31]	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	89.3

Table 1: **Average Precision (%) on VOC 2007 test set.** The entries with the highest APs for each object category are printed in bold. The top four methods use VGG16 features where LSSVM-Max is for max pooling and LSSVM-SSD learns SSD based on LSSVM scores. RRSVM has the highest AP in all categories.

ods in **every** object category especially in the ones where the baseline classifiers have relatively low accuracy (e.g., for categories bottle and plant, the performance gaps are 10.2% and 9.2% respectively). Some previous state-of-the-art methods are listed at the bottom of Tab. 1, and RRSVM outperforms them by a wide margin (approximately 30% error reduction).

Figure 2 shows the local regions used by RRSVM on some representative test images. From these images we see that RRSVM assigns weights to meaningful regions of the images that contain relevant object information. More examples are provided in the supplemental material.

4.3. VOC 2012 Action Classification

In this section, we describe the experiments on the *Action dataset* from the PASCAL VOC2012 Challenge. The task is to recognize actions in still images, given the bounding box of a person performing the action. The dataset contains 2296 training images with 3134 bounding boxes, 2292 validation images with 3144 bounding boxes, and 4569 test images with 6283 bounding boxes.

This classification task is similar to the image classification task of VOC2007 described in the previous section. The key difference is the availability of the regions of interest, which are the human bounding boxes. Following [12, 14, 31], for each method in consideration, we train two classifiers, one for the entire images and the other for the human regions. Finally, we combine two classifiers by averaging their scores.

Table 2 compares the performance of RRSVM and LSSVM. Both methods use the same sets of feature vectors produced by VGG16. As can be seen, RRSVM outperforms LSSVM on all action categories. The gap in mean average precision is 3.2%.

Table 3 compares RRSVM with several state-of-the-art methods on the test set. For this experiment, we use both VGG16 and VGG19 as also used by [31]. We train the RRSVMs for VGG16 and VGG19 separately and average their scores. We train them on both train and validation sets. The results on the test set are obtained by submitting the output to the PASCAL evaluation server, conforming to the rules of PASCAL VOC Challenge. As can be seen in Table 3, our proposed RRSVM, without any task-specific heuristics, outperforms the previous state-of-the-art methods.

4.4. ILSVRC 2014 Image Classification

We also evaluate RRSVM on ILSVRC 2014 classification challenge [27] to study its benefits for large-scale image classification. The ILSVRC 2014 classification challenge requires classifying images into one of the 1000 leaf-node categories of the ImageNet dataset. There are 1,281,167 images for training, 50,000 for validation, and 100,000 for testing. The number of positive training examples for each class ranges from 732 to 1300, and all images of the other classes are considered as negative examples. Two numbers are usually reported: the top-1 error rate, which compares the class with highest prediction score with the ground truth class, and the top-5 error rate, which considers the prediction correct if the ground truth is within the top 5 predicted classes. The top-5 error is used to address the potential problem of inexhaustive ground truth annotation.

4.4.1 Experiment setup

The experiment setup for the ILSVRC dataset is similar to the setup for the PASCAL VOC 2007 dataset (Section 4.2) with two modifications to: 1) be possible to train 1000 classifiers on a large dataset of more than one million images, and 2) calibrate the scores of 1000 binary one-versus-rest classifiers for a multi-class categorization task.

The first modification is to train LSSVM and RRSVM with a smaller number of negative examples. In theory, it is possible to train LSSVM and RRSVM with all the ex-

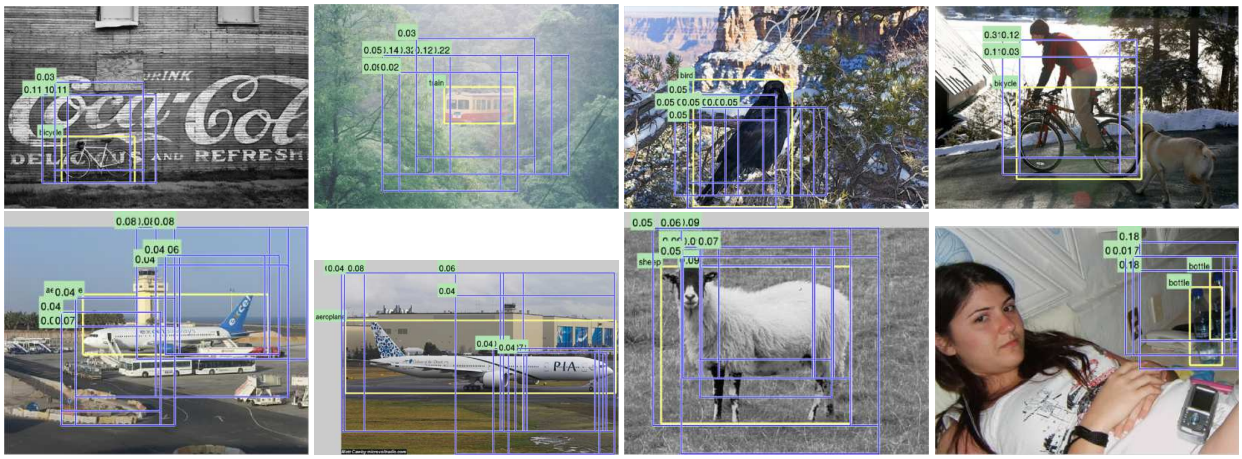


Figure 2: **Local regions used by RRSVM for classification.** These are representative images from the test set of PASCAL VOC 2007. The blue boxes with numbers correspond to the regions and their weights used for classification. Yellow boxes with labels are the ground truth regions of interest for the specific category. Best viewed on a digital device.

Model	jump.	phone.	playinstr.	read.	ridebike	ridehorse	run	takephoto	usecomp.	walk.	mAP
LSSVM-Mean	85.5	64.5	95.2	71.6	93.0	95.9	82.5	69.0	90.7	62.0	81.0
LSSVM-Max	85.2	65.3	95.8	73.3	93.4	96.5	84.3	70.2	90.0	61.3	81.5
LSSVM-SSD	85.2	65.7	95.8	73.2	93.6	96.5	84.4	70.1	90.0	62.3	81.7
RRSVM	**87.2**	**72.3**	**96.2**	**76.9**	**94.6**	**96.8**	**87.8**	**74.4**	**92.9**	**62.6**	**84.2**

Table 2: **Average precision (%) on VOC2012 Action validation set**. Both methods use VGG16 features. LSSVM average all feature vectors while RRSVM uses them selectively. RRSVM outperforms LSSVM on all action categories.

amples. In practice, however, this creates a bottleneck in data loading because the data is too big to be fit in memory, and this is especially problematic for iterative optimization. To circumvent this problem, we train LSSVM and RRSVM with a reduced set of negative examples. The set of negative examples is based on hard negative mining as follows. For each class, we first train an LSSVM with all positive training examples and a random set of negative examples. To maximize the diversity, we randomly pick 15 negative examples in each of the 999 negative classes. The LSSVM obtained is subsequently used to select a new set of hard negative examples for retraining the LSSVM. The new set of hard negative examples is selected from negative examples with the highest classification scores. Additionally, we maintain the cardinality of the negative set and ensure that each negative class has at least 10 examples. The procedure for training an LSSVM and picking a new set of negative examples is repeated for three iterations (it often converges in three iterations). The final selected set of negative examples is then used to train both an LSSVM and an RRSVM.

The second modification is to integrate multiple binary classifiers in to a single multi-class classifier. Since RRSVMs (and also LSSVMs) are independently trained for each class, they must be calibrated to be used for multi-class classification. We use a 3-layer neural network to reconcile the outputs of the 1000 classifiers. The first layer of the network is a fully connected layer with 1000 units followed by a rectified linear unit (ReLU). The second layer is the same as the first layer but without ReLU. The last layer is a softmax layer with 1000 outputs for 1000 classes. The network is trained to minimize the negative log likelihood using all training examples.

Method	mAP
Oquab et al. [24]	70.2
Hoai et al. [14]	70.5
Gkioxari et al. [10]	73.6
Hoai [12]	76.3
Simonyan and Zisserman [31]	84.0
RRSVM (ours)	**85.5**

Table 3: **Mean average precision on VOC2012 action recognition test set.** RRSVM outperforms the previous state-of-the-art by 1.5%.

4.4.2 Classification performance

Table 4 compares the performance of RRSVM with several other methods on the validation and the test sets of the ILSVRC 2014 dataset. Overall, the results of RRSVM are comparable with the state-of-the-art methods even though it uses only **one** neural network and without the expensive multi-cropping procedure. The fairest comparison is between RRSVM and VGG16 that does not use multi-crop (first method of Table 4) because both methods use the same features and have the same setup. In this comparison, RRSVM outperforms the baseline by 0.8% and 1.9% using Top-5 and Top-1 error measurement respectively.

4.4.3 Category based analysis

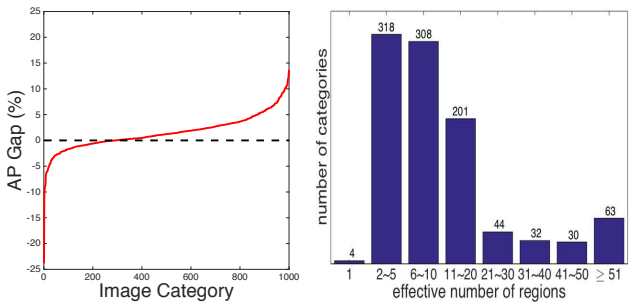
The diversity and size of ILSVRC 2014 enable us to further analyze the performance of RRSVM. In this section we use AP to measure the performance on the validation set for individual object categories.

We first of all consider the differences between APs of RRSVM and results of VGG16, which will be referred to as AP gaps. We sort the AP gaps for 1000 classes and plot them in Figure 3a. As can be seen, the AP gaps are positive for more than70% of the classes. The mean of the AP gaps is 1.6%. The category with the largest AP gap is n03673027 (ocean liner) with 13% AP gap. The lowest AP gap is -23.7% for n02447721 (gong), which seems to be an outlier case as the second lowest AP gap is -9.8%. There are two possible reasons for poor performance on this category: (i) the images of gong have various shapes/textures and with relatively uniform backgrounds, and the classifier being trained by RRSVM deteriorates through the iterations of the iterative optimization; and (ii) we did not use enough negative training examples (recall we only use 15K negative training samples for efficiency). Examples of the failure cases are showed in the supplementary material.

We also consider the number of non-zero values of the s vector. It corresponds to the number of local regions used by RRSVM. Figure 3b plots the distribution of the numbers of effective regions. RRSVM uses no more than 10 regions for the majority of the categories.

4.5. Timing

The optimization of RRSVM typically converges within 15 iterations. Each iteration requires solving a QCQP and one or several least square problems. On an Intel Xeon 2.5GHz machine with 16 cores for parallel data loading, it takes roughly 10 minutes to train an RRSVM classifier for each ILSVRC 2014 category. This excludes the feature computation using VGG16.



(a) (b)

Figure 3: (a): AP gaps between RRSVM and VGG16 for individual classes on ILSVRC 2014. The majority of classes have positive AP gaps. (b): distribution of the numbers of effective regions used by RRSVM on ILSVRC 2014. Most classes require 20 effective regions or less.

	Validation		Test
Model	top-1	top-5	top-5
VGG16, dense eval. (baseline)	24.8	7.5	-
VGG16, dense eval., multi-crop	24.4	7.1	7.0
VGG16+19, dense eval., multi-crop	23.7	6.8	6.8
GoogLeNet (7 nets, multi-crop)	-	6.7	**6.7**
RRSVM	**22.9**	**6.7**	6.8

Table 4: **Error rates on ILSVRC 2014 dataset (%)**. RRSVM performs similarly to methods that use multiple nets and use multiple cropped regions (computationally expensive). RRSVM uses the same setup as the first method shown in this table.

5. Conclusions

We have proposed Region Ranking SVM (RRSVM), a novel framework for image classification. RRSVM is developed on the foundation of a robust decision function that is based on the distribution of multiple region scores. During training, RRSVM jointly learns a region evaluation function and a region-combination vector. We have showed that RRSVM outperforms the baseline methods that use the same features as RRSVM by a wide margin. Tested on the image classification task of PASCAL VOC 2007, the action recognition task of PASCAL VOC 2012, and the object classification task of ILSVRC 2014, RRSVM is better than or comparable to the state-of-the-art methods that use more powerful features or require more computational resources.

Currently, each RRSVM is a binary classifier, and multiple RRSVMs must be calibrated before they can be used for multi-class classification. A future direction is to extend the current framework to jointly train multiple RRSVMs and the multi-class classifier.

Acknowledgment. This project is partially supported by the National Science Foundation Award IIS-1566248.

References

[1] Z. Akata, F. Perronnin, Z. Harchaoui, and C. Schmid. Good practice in large-scale learning for image classification. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(3), 2014. 2

[2] A. Bosch, A. Zisserman, and X. Munoz. Representing shape with a spatial pyramid kernel. In *Proceedings of the ACM International Conference on Image and Video Retrieval*. ACM, 2007. 2

[3] K. Chatfield and A. Zisserman. Visor: Towards on-the-fly large-scale object category retrieval. In *Proceedings of the Asian Conference on Computer Vision*. Springer, 2013. 2

[4] K. Chatfield, K. Simonyan, A. Vedaldi, and A. Zisserman. Return of the devil in the details: Delving deep into convolutional nets. *arXiv preprint arXiv:1405.3531*, 2014. 1, 2, 6

[5] D. Chen, X. Cao, F. Wen, and J. Sun. Blessing of dimensionality: High-dimensional feature and its efficient compression for face verification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 2013. 2

[6] P.-M. Cheung and J. T. Kwok. A regularization framework for multiple-instance learning. In *Proceedings of the International Conference on Machine Learning*, 2006. 2

[7] M. Everingham, L. Van Gool, C. K. Williams, J. Winn, and A. Zisserman. The pascal visual object classes (voc) challenge. *International Journal of Computer Vision*, 88(2), 2010. 1, 5

[8] M. Everingham, L. Van Gool, C. Williams, J. Winn, and A. Zisserman. The PASCAL Visual Object Classes Challenge 2012 (VOC2012) Results. www.pascal-network.org/challenges/VOC/voc2012/workshop/, 2012. 5

[9] P. V. Gehler and O. Chapelle. Deterministic annealing for multiple-instance learning. In *Proceedings of the International Conference on Artificial Intelligence and Statistics*, 2007. 4

[10] G. Gkioxari, R. Girshick, and J. Malik. Actions and attributes from wholes and parts. *arXiv preprint arXiv:1412.2604*, 2014. 7

[11] K. Grauman and T. Darrell. The pyramid match kernel: Discriminative classification with sets of image features. In *Proceedings of the International Conference on Computer Vision*, volume 2. IEEE, 2005. 2

[12] M. Hoai. Regularized max pooling for image categorization. In *Proceedings of the British Machine Vision Conference*, 2014. 5, 6, 7

[13] M. Hoai and A. Zisserman. Improving human action recognition using score distribution and ranking. In *Proceedings of the Asian Conference on Computer Vision*, 2014. 2, 3

[14] M. Hoai, L. Ladicky, and A. Zisserman. Action recognition from weak alignment of body parts. In *Proceedings of the British Machine Vision Conference*, 2014. 6, 7

[15] M. Hoai, L. Torresani, F. De la Torre, and C. Rother. Learning discriminative localization from weakly labeled data. *Pattern Recognition*, 47:1523–1534, 2014. 2

[16] Y. Hu, M. Li, and N. Yu. Multiple-instance ranking: Learning to rank images for image retrieval. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2008. 2

[17] H. Jégou, F. Perronnin, M. Douze, J. Sanchez, P. Perez, and C. Schmid. Aggregating local image descriptors into compact codes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(9), 2012. 2

[18] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems*, 2012. 1, 2

[19] M. P. Kumar, B. Packer, and D. Koller. Self-paced learning for latent variable models. In *Advances in Neural Information Processing Systems*, 2010. 4

[20] S. Lazebnik, C. Schmid, and J. Ponce. Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, volume 2. IEEE, 2006. 2

[21] M. Lin, Q. Chen, and S. Yan. Network in network. *arXiv preprint arXiv:1312.4400*, 2013. 2

[22] Y. Lin, F. Lv, S. Zhu, M. Yang, T. Cour, K. Yu, L. Cao, and T. Huang. Large-scale image classification: fast feature extraction and svm training. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 2011. 2

[23] M. H. Nguyen, L. Torresani, F. De la Torre, and C. Rother. Weakly supervised discriminative localization and classification: a joint learning process. In *Proceedings of the International Conference on Computer Vision*, 2009. 2

[24] M. Oquab, L. Bottou, I. Laptev, and J. Sivic. Learning and transferring mid-level image representations using convolutional neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 2014. 7

[25] F. Perronnin, J. Sánchez, and T. Mensink. Improving the fisher kernel for large-scale image classification. In *Proceedings of the European Conference on Computer Vision*, 2010. 2

[26] H. Pirsiavash, D. Ramanan, and C. Fowlkes. Bilinear classifiers for visual recognition. In *Advances in Neural Information Processing Systems*, 2009. 2

[27] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision*, pages 1–42, April 2015. doi: 10.1007/s11263-015-0816-y. 1, 5, 6

[28] J. Sánchez and F. Perronnin. High-dimensional signature compression for large-scale image classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 2011. 2

[29] C. Saunders, A. Gammerman, and V. Vovk. Ridge regression learning algorithm in dual variables. In *Proceedings of the International Conference on Machine Learning*, 1998. 3

[30] P. Sermanet, D. Eigen, X. Zhang, M. Mathieu, R. Fergus, and Y. LeCun. Overfeat: Integrated recognition, localization and detection using convolutional networks. *arXiv preprint arXiv:1312.6229*, 2013. 1, 2, 5

[31] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014. 1, 2, 5, 6, 7

[32] P. Siva, C. Russell, and T. Xiang. In defence of negative mining for annotating weakly labelled data. In *Proceedings of the European Conference on Computer Vision*, 2012. 4

[33] J. A. K. Suykens and J. Vandewalle. Least squares support vector machine classifiers. *Neural Processing Letters*, 9(3): 293–300, 1999. 3

[34] J. A. K. Suykens, T. V. Gestel, J. D. Brabanter, B. DeMoor, and J. Vandewalle. *Least Squares Support Vector Machines*. World Scientific, 2002. 3, 5

[35] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. *arXiv preprint arXiv:1409.4842*, 2014. 1, 2, 5

[36] J. B. Tenenbaum and W. T. Freeman. Separating style and content with bilinear models. *Neural Computation*, 12(6): 1247–1283, 2000. 2

[37] V. Vapnik. *Statistical Learning Theory*. Wiley, New York, NY, 1998. 3

[38] A. Vedaldi and K. Lenc. Matconvnet – convolutional neural networks for matlab. 2015. 5

[39] T. Y. Vicente, M. Hoai, and D. Samaras. Leave-one-out kernel optimization for shadow detection. In *Proceedings of the International Conference on Computer Vision*, 2015. 3, 5

[40] Y. Wei, W. Xia, J. Huang, B. Ni, J. Dong, Y. Zhao, and S. Yan. Cnn: Single-label to multi-label. *arXiv preprint arXiv:1406.5726*, 2014. 6

[41] J. Yang, K. Yu, Y. Gong, and T. Huang. Linear spatial pyramid matching using sparse coding for image classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 2009. 2

[42] J. Ye and T. Xiong. Svm versus least squares svm. In *Proceedings of the International Conference on Artificial Intelligence and Statistics*, 2007. 5

[43] M. D. Zeiler and R. Fergus. Visualizing and understanding convolutional networks. In *Proceedings of the European Conference on Computer Vision*. Springer, 2014. 2

[44] X. Zhou, K. Yu, T. Zhang, and T. S. Huang. Image classification using super-vector coding of local image descriptors. In *Proceedings of the European Conference on Computer Vision*. Springer, 2010. 2