Backprojection Revisited: Scalable Multi-view Object Detection and Similarity Metrics for Detections

Nima Razavi[1], Juergen Gall[1], and Luc Van Gool[1,2]

[1] Computer Vision Laboratory, ETH Zurich
[2] ESAT-PSI/IBBT, KU Leuven
nrazavi|gall|vangool@vision.ee.ethz.ch

Abstract. Hough transform based object detectors learn a mapping from the image domain to a Hough voting space. Within this space, object hypotheses are formed by local maxima. The votes contributing to a hypothesis are called support. In this work, we investigate the use of the support and its backprojection to the image domain for multi-view object detection. To this end, we create a shared codebook with training and matching complexities independent of the number of quantized views. We show that since backprojection encodes enough information about the viewpoint all views can be handled together. In our experiments, we demonstrate that superior accuracy and efficiency can be achieved in comparison to the popular one-vs-the-rest detectors by treating views jointly especially with few training examples and no view annotations. Furthermore, we go beyond the detection case and based on the support we introduce a part-based similarity measure between two arbitrary detections which naturally takes spatial relationships of parts into account and is insensitive to partial occlusions. We also show that backprojection can be used to efficiently measure the similarity of a detection to all training examples. Finally, we demonstrate how these metrics can be used to estimate continuous object parameters like human pose and object's viewpoint. In our experiment, we achieve state-of-the-art performance for view-classification on the PASCAL VOC'06 dataset.

1 Introduction

As an important extension of the Generalized Hough Transform (GHT) [2], the Implicit Shape Model (ISM) [3] trains a codebook of local appearance by clustering a training set of sparse image features and storing their relative location and scale with respect to the object center. For detection, sparse image features are extracted from a test image and are matched against the codebook casting probabilistic votes in the 3D Hough accumulator as illustrated in Fig. 1(a,b). The support is composed of all the votes that contribute to a detection. The backprojection of the support gives direct evidence of the object's presence and can be used as hypothesis verification, Fig. 1(c,d). For instance, by providing pixel-accurate segmentation of the training data and storing this information in the codebook, the backprojection can be augmented by top-down segmentation [3].

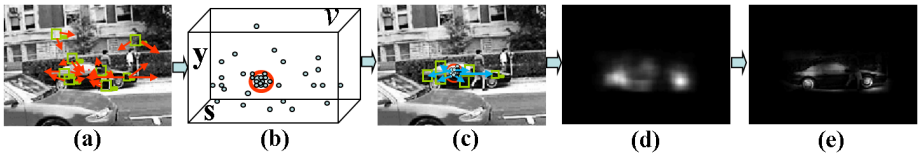


Fig. 1. Object detection using Implicit Shape Models: (a) Features are matched against the codebook $\mathcal{C}$ casting votes to the voting space $\mathcal{V}$. (b) The local maxima of the voting space is localized and the votes contributing to it are identified (inside the red circle). (c) The votes are backprojected to the image domain creating the backprojection mask. (d-e) Visualization of the backprojection mask. Note that the mask does not include the area occluded by a pedestrian. The image is taken from UIUC cars dataset [1].

Although the support and its backprojection have been used for verification and meta-data transfer [3,4], it has not yet been fully explored. In this work, we address a broader question: what does the support tell us about the detection? We show that additional properties of an object can be retrieved from the support without changing the training or the detection procedure. To this end, we augment the codebook by some additional information to establish a link between the detection and the parts of different training examples. In particular, we demonstrate two important properties of the support:

Firstly, the different views of an object can be handled by a single codebook and even a single view-independent voting space since the support and its backprojection encode enough information to deal with the viewpoint variations. This is very relevant in practice since state-of-the-art GHT-based multi-view detectors like [5,6] treat different viewpoints as different classes and train a battery of one-vs-the-rest codebooks for each view. The training and detection time of these approaches scale thus linearly with the number of quantized views. Unlike these approaches, we train a single shared codebook for all views, i.e. the complexity of training and matching a feature against it is independent of the number of quantized views. The proposed training procedure also allows us to make better use of training data by sharing features of different views. Having the shared codebook and depending on the availability of view annotations and amount of training data, two voting schemes are proposed which outperform the battery of one-vs-the rest detectors both in terms of accuracy and computational complexity, in particular, when only few training examples are available.

Secondly, not only we can detect objects under large view variations with a single shared codebook, bu also we can use the support for defining similarity measures to retrieve nearest examples. One can then estimate continuous parameters (like object's pose or view) of detections by the parameters of the nearest examples. To this end, we introduce two similarity measures based on the support. The first one efficiently measures the similarity of a detection to all training examples. We show that this measure is applicable to retrieve various continuous parameters like the pose of a detected pedestrian. The second metric finds dense

feature correspondences and can be used as a similarity measure between any two detections. This measure is particularly interesting as it is part-based and naturally takes the spatial relationships of the parts into account. It also inherits nice properties like good generalization and insensitivity to partial occlusions from the ISM model. The power of the latter similarity metric is demonstrated in the task of view-retrieval in presence of partial occlusions.

2 Related Work

Several approaches have been proposed for the creation of codebooks and to learn the mapping of image features into the Hough space. While [3] clusters the sparse image features only based on appearance, the spatial distribution of the image features is also used as a cue for the clustering in [7,8]. [9] proposes to store training features without clustering and use them as a codebook. Hough forests [10] use a random forest framework instead of clustering for codebook creation.

In order to vote with a codebook, matched codewords cast weighted votes. While the work of [3] uses a non-parametric Parzen estimate for the spatial distribution and thus for the determination of the weights, [11] re-weights the votes using a max-margin framework for better detection. The voting structure has been addressed in [12], where voting lines are proposed to better cope with scale-location ambiguities.

In [3], the backprojection has been used for verification. To this end, the training data is segmented and the local foreground-background masks are stored with the codebook. When a maximum is detected in the voting space, the local segmentation masks are used to infer a global segmentation for the detection in the image. The global segmentation is then in turn used to improve recognition by discarding votes from background and reweighting the hypothesis. In [4], the individual parts of an object (e.g. front wheel of a motorbike) are also annotated in the training and used to infer part-labels for a test instance. In this work, we neither focus on hypothesis verification nor require time consuming segmentations and part annotations of the training data.

The handling of multiple object views has also been addressed in the literature, see e.g. [13]. Based on the ISM and a silhouette-based verification step [3], the model can be extended by handling the viewpoint as additional dimension. Thomas et al. [5] train a codebook for each annotated view and link the views together by appearance. [14] extends the voting scheme of ISM by a 4th dimension, namely shape represented by silhouettes, to improve the segmentation based verfication step. Although this approach uses a shared codebook for multi-aspect detection of pedestrians, it is limited as it only considers pedestrians which are already handled by the ISM. Other approaches use the statistical or geometric relationships between views in the training data to reason about the 3D structure of the object [15,16,17,18]. Since these approaches need many viewpoints of an object, they are very expensive in data. Although some of the missing data can be synthesized from existing data by interpolation [16,19], the interpolation still

requires a certain number of views in the training data. In [20], a discriminative approach is proposed to handle aspects more general than for a specific class. To this end, latent variables that model the aspect are inferred and the latent discriminative aspect parameters are then used for detection.

3 Multi-view Localization with ISMs

The object detection in our approach is based on the Implicit Shape Model (ISM) [3]. In this framework, training consists of clustering a set of training patches P^{train} to form a codebook of visual appearance $\mathcal{C}$ and storing patch occurrences for each codebook entry. At runtime, for each test image $\mathcal{I}_{test}$, all test patches P^{test} are extracted and matched against $\mathcal{C}$. Each patch casts weighted votes for the possible location of the object center. These votes are based on the spatial distribution of the matching codebook entry to a Hough accumulator $\mathcal{V}$. This encodes the parameters of the object center, e.g. position and scale in the image domain; see Fig. 1(a). This way all the votes are cast in $\mathcal{V}$ (Fig. 1(b)). The probability of the presence of the object center at every location of $\mathcal{V}$ is estimated by a Parzen-window density estimator with a Gaussian kernel. Consistent configurations are searched as local maxima in $\mathcal{V}$ forming a set of candidates $\mathbf{v}_h \in \mathcal{V}$. For each candidate $\mathbf{v}_h$, its support S_h is formed by collecting all the votes contributing to its detection.

In the following, we discuss the training of the shared codebook for all the views in detail and explain the necessary augmentation of the codebook entries. Then multi-view detection with this codebook is discussed. Afterwards, we look into the support, S_h, and its backprojection to the image domain for bounding box estimation and retrieving nearest training examples. Finally, we introduce a similarity metric based on the support for comparing two arbitrary detections.

3.1 Training a Shared Codebook

To train the codebook $\mathcal{C}$ with entries $c_1 \dots c_{|\mathcal{C}|}$, a set of training patches are collected P^{train}. Training patches are sampled from a set of bounding box annotated positive images and a set of background images. Each training patch, $P_k^{train} = (\mathcal{I}_k, l_k, \mathbf{d}_k, \theta_k)$, has an appearance $\mathcal{I}_k$, class label l_k, a relative position to the object center and an occurrence scale $\mathbf{d}_k = (x_d, y_d, s_d)$, and additional training data information θ_k, e.g. the identity of the training image it is sampled from.

For building the codebook, we use the recently developed method of Hough Forests [10] as it allows us to use dense features and also due to its superior performance compared to other methods, e.g. the average-link clustering used in [3]. Hough Forests are random forests [21] which are trained discriminatively to boost the voting performance. During training, a binary test is assigned recursively to each node of the trees that splits the training patches into two sets. Splitting is continued until the maximum depth is reached or the number of remaining patches in a node is lower than a predefined threshold (both fixed to

20 in the current implementation). The codebook consists of the leaves which store the arrived training patches P_k^{train}.

The binary tests are selected using the same two optimization functionals as [10], to reduce the uncertainty of class-labels and offset. The view annotations are completely discarded. In our implementation, the label of each patch, l_k, is a binary value assigned to one if the patch was drawn from inside the bounding box of a positive image and otherwise it is set to zero. However, all training data information, including the label, can be recovered from θ_k.

3.2 Multi-view Detection

For detection of objects in a test instance, every patch of the test instance P_i^{test} is matched against the codebook $\mathcal{C}$ and its probabilistic votes are cast to the voting space $\mathcal{V}$. In particular, by matching the patch $P_i^{test} = (\mathcal{I}_i, x_i, y_i, s_i)$ to the codebook, the list of all occurrences $\mathcal{O}_i = \{o = (\mathcal{I}, l, \mathbf{d}, \theta)\}$ stored in the matching entries is obtained. In this paper, we propose two schemes for casting these votes to $\mathcal{V}$: ***joint voting*** and ***separate voting***.

Joint Voting: Votes are cast to a 3D voting space $\mathcal{V}(x, y, s)$. Let us denote the proportion of positive (same label) to negative (different label) patches of each codebook entry by r_c^{pos} and the number of positive occurrences by n_c^{pos}. Then for each occurrence $o = (\mathcal{I}, l, \mathbf{d}, \theta) \in \mathcal{O}_i$, a vote with weight w_o is cast to the position $\mathbf{v} = (v_1, v_2, v_3)$:

$$v_1 = x_i - x_d(s_i/s_d) \tag{1}$$
$$v_2 = y_i - y_d(s_i/s_d) \tag{2}$$
$$v_3 = s_i/s_d \tag{3}$$
$$w_o = r_c^{pos}/n_c^{pos} \tag{4}$$

After all votes are cast, the local maxima of $\mathcal{V}$ are found, forming a set of candidate hypotheses. In this voting scheme, votes from different training examples and from different views can contribute to a hypothesis and detection is performed without using the viewpoint annotations.

Separate Voting: Voting in ***separate voting*** is performed in a 4D voting space $\mathcal{V}(x, y, s, view)$. For each view $view$, let us denote the number and the proportion of positive occurrences in c with label $view$ by n_c^{view} and r_c^{view}, respectively. Then for each occurrence o, a vote is cast to the position $\mathbf{v} = (v_1, v_2, v_3, v_4)$ with weight w_o:

$$v_1 = x_i - x_d(s_i/s_d) \tag{5}$$
$$v_2 = y_i - y_d(s_i/s_d) \tag{6}$$
$$v_3 = s_i/s_d \tag{7}$$
$$v_4 = view \tag{8}$$
$$w_o = r_c^{view}/n_c^{view} \tag{9}$$

The local maxima of $\mathcal{V}$ are found after all votes are collected to form a set of candidate hypotheses. In this voting scheme, only votes from training examples of a particular viewpoint can contribute to a hypothesis of that view.

3.3 Detection Support and Backprojection

After the local maxima are localized in $\mathcal{V}$, the candidate hypotheses are determined in terms of their center position and scale (and view in separate voting). We then collect all votes in the support of a hypothesis (i.e. votes contributing to its detection) and exploit it to get additional information about it. For a hypothesis h in location $\mathbf{v}_h \in \mathcal{V}$, we define its support S_h as (see Fig. 1(b)):

$$S_h = \{\mathbf{v} \in \mathcal{V} | K(\mathbf{v} - \mathbf{v}_h) > 0\} \ . \tag{10}$$

where K is a radially symmetric (in x and y) kernel with only local support such that the set S_h contains only votes in the local neighborhood of $\mathbf{v}_h$. In the current implementation, only votes from the same scale, and same view in the case of separate voting, are considered as votes contributing in the support.

Additionally, we can define the backprojection as a mapping from $\mathcal{V}$ to the image domain to form the backprojection mask $\mathcal{M}$ (see Fig. 1(c)):

$$B : \{\mathbf{v} \in \mathcal{V} : \mathit{condition}\} \mapsto \mathcal{M} \ . \tag{11}$$

where *condition* are constraints on the votes. Having a constraint, e.g. $\mathbf{v} \in S_h$, the mask is constructed by projecting all the votes satisfying the constraint back to the image domain. Since each feature point $\mathbf{x} = (x, y)$ in the test image is mapped to the voting space for detection, the mask can be calculated by mapping every vote $\mathbf{v} = (v_1, v_2, v_3)$ back to $\mathbf{x}$ with weight w_o (see Fig. 1(d) for an example of such a mask). The total weight of a mask $w_{\mathcal{M}}$ is then defined as the sum of the weights of all the votes mapped to it.

Bounding box estimation from backprojection: Most approaches (e.g. [3,6,10]) estimate the extent of the object's presence by placing the average bounding box of training images scaled and translated to the detection center. Although this measure is sufficiently accurate for the rather generous standard evaluation criteria like [22], this measure is not applicable to multi-view detection with joint voting where aspect ratios of different views widely vary. Inspired by [3], we propose using the backprojection of the supporting features for this purpose. In our work, the backprojection mask is simply thresholded by an adaptive threshold (set to half the value range) to form a binary mask. The tightest bounding box encompasing this mask is used as our bounding box estimate. Of course this is an oversimplification and there is still the possibility of more sophisticated bounding box estimations, e.g. [23], but simple thresholding suffices to obtain reasonable bounding box estimates.

Retrieving nearest training images: By conditioning the back-projection of a hypothesis support S_h to the votes coming from a single training example with identity tr, one can measure how much tr contributes to the detection of h. Formally, we can write

$$B : \{\mathbf{v} \in \mathcal{V} : \mathbf{v} \in S_h \wedge \theta_{\mathbf{v}} = tr\} \mapsto \mathcal{M}_{tr} \ . \tag{12}$$

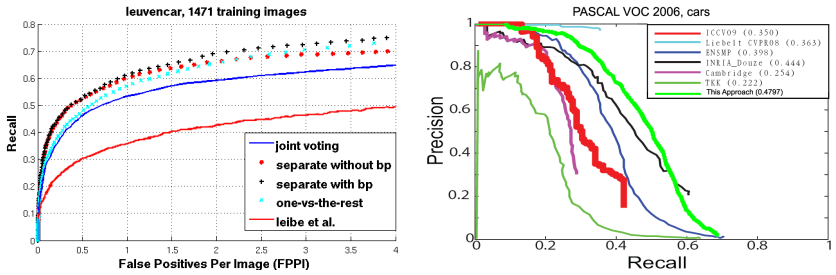
The total weight of $\mathcal{M}_{tr}$, $w_{\mathcal{M}_{tr}}$, can then be used as a holistic measure of similarity between the hypothesis h and the training image tr. In principle, by introducing additional constraints, one can enforce more specific similarity measures, e.g. similarity of the left side of h to the left side of tr. Since we sample only a sparse set of patches from the training examples during training, this measure establishes correspondences between sparse locations of the detection and a training example. Fig. 7 shows some examples of $\mathcal{M}_{tr}$.

Support intersection as a metric: Let $I(o, S_h)$ be an indicator variable which is one if there is any vote in S_h which comes from occurrence $o \in \mathcal{O}$ and zero otherwise. We define the support intersection of two hypotheses h_1 and h_2 as:

$$S_{h_1} \cap S_{h_2} = \frac{\sum_{o \in \mathcal{O}} w_o I\left(o, S_{h_1}\right) I\left(o, S_{h_2}\right)}{\sum_{o \in \mathcal{O}} w_o I(o, S_{h_1})} \ . \tag{13}$$

Note that the similarity measure is not symmetric due to the normalization factor. Yet, this factor is important as it makes the measure independent of the detection weight and as it can also account for occluded regions. The support intersection can be used as a similarity measure between two detections. This similarity measure is a model-based similarity measure. There is a close link between the support intersection in (13) and the histogram intersection kernel used in bag-of-words image classification [24]. This said, there are also substantial differences between the two. Since the detection is done with ISM, the support of the detection takes the spatial relationships of the features into account. Therefore, there is no need for a fixed grid on top of the bag-of-words representation as in the spatial pyramid kernel [24]. In addition, this metric is part-based and benefits from the generalization capabilities of part-based methods and their insensitivity to occlusions, as shown in the experiments.

It is worthwhile to note an important difference between the similarity measures (12) and (13). The similarity measure in (12) can only be used to find the similarity between sparse patches of a detection and a training example, i.e. only matching to the same patches sampled during training. But support intersection establishes a dense feature correspondence between any two detections. Due to the dense correspondences in (13), for comparing two detections, this similarity measure has a computational cost in the order of the number of votes in its support. However, this is about the same cost it takes to consider sparse correspondences to all training examples in (12).



(a) (b)

Fig. 2. (a) Detection performance for the Leuven-cars dataset and comparison to Leibe et al. [6]. Separate voting with bounding boxes estimated from backprojection (bp) achieves the best performance despite much lower training and detection complexity than baseline (one-vs-the-rest). Joint voting with even lower detection complexity and without using view annotations gives competitive results. (b) Performance comparison of joint-voting with state-of-the-art approaches on PASCAL VOC 2006 cars dataset.

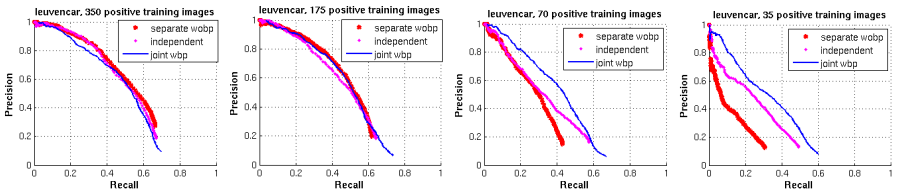
4 Experiments

In order to assess the performance of the multi-view detectors described in Sect. 3.2, we use three datasets. The multi-view Leuven-cars dataset [6] contains 1471 training cars annotated with seven different views and a sequence of 1175 images for testing. The multi-view Leuven-motorbikes dataset [5] contains 217 training images annotated with 16 quantized views and 179 test images. And the PASCAL VOC'06 cars datset [22]. Further experiments for nearest neighbor retrieval are carried out on the TUD-pedestrians dataset introduced in [25] and the cars datasets. The TUD-pedestrians dataset provides 400 training images and 250 images for testing. Throughout the experiments, only bounding box annotations of the training images are used. The segmentation masks that are provided for some datasets are discarded.

4.1 Multi-view Detection

As a baseline comparison for the multi-view detection, we consider the popular one-vs-the-rest detector. For each view, the training is carried out with the positive training images of a view versus random patches from the Caltech 256 clutter set plus all the positive training images of the other views. An edge detector has been carried out both for training and testing and only features with their center on an edge are considered.

In order to make fair comparisons, the training and detection parameters are kept the same throughout the experiments. In particular, the number of trees in each forest is set to 15. From each training image 100 patches are sampled and the number of background patches is kept constant at 20000 patches. For detection, the kernel used for the density estimation is a Gaussian with $\sigma = 2.5$ and the first 20 local maxima per image are considered. When the backprojection is not used for bounding box estimation, non-maxima suppression is done by



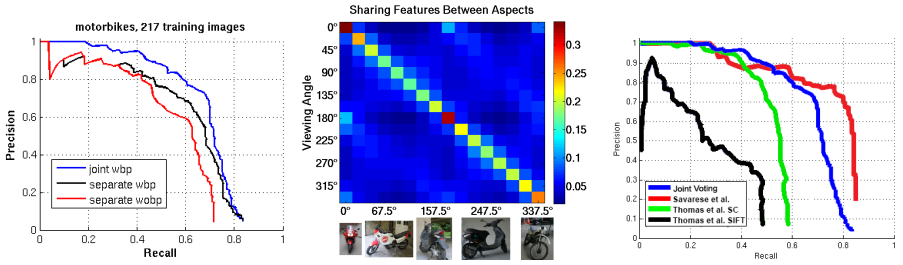
(a) 350 images (b) 175 images (c) 70 images (d) 35 images

Fig. 3. The effect of training size on the performance of joint voting, separate voting, and a battery of independent one-vs-the-background classifiers: With the abundance of training data per view, the separate-voting works best. The advantage of sharing is significant with lower number of training examples, especially compared to the separate voting with an identical codebook although no view annotations are used. Joint and separate voting outperform the independent detector in efficiency and/or accuracy.

removing all detections whose centers are within the bounding box (with 90% of its size) of another detection with a higher weight. When using backprojection, the hypothesis with the highest weight is included and its features are removed from all other hypotheses, thereby decreasing their weights.

The results of this experiment are shown in Fig. 2. As can be seen, separate voting with the help of backprojection performs best and estimating the bounding box with backprojection slightly increases the performance of the system. Joint voting also shows a competitive performance. It is worthwhile to note that the superior performance of separate voting is mainly due to the abundance of training images per view in this dataset and the presence of additional view information not used by joint voting. By sharing features across views, as e.g. shown in the work of Torralba et al. [26], one expects to benefit mainly when the training data is limited. In order to verify this, we did the following experiment.

We compare the performance of joint voting, separate voting, and a battery of independent one-vs-the-background classifiers for 50, 25, 10, and 5 training images per view (7 views). In all the experiments, the full background set from Caltech 256 clutter is used and the set of training images for all three detectors is identical. Joint voting and separate voting use identical shared codebooks whereas a separate codebook is trained per view for the independent detector (see Fig. 3). With fewer training examples, as expected, the performance of all three detectors degrades, but that of joint voting far more gently. In particular, the comparison of separate voting and joint voting for few training images is very interesting. Although an identical codebook is used, joint voting significantly outperforms separate voting. This performance gap seems to narrow by using several codebooks (number of views) and thus more codebook entries for the independent detector but the performance of joint voting is still superior in terms of accuracy as well as training and detection time. In order to assess performances on a more challenging dataset, we evaluated joint voting and separate voting for the Leuven-motorbikes dataset [5] where the test set is provided by the PASCAL VOC Challenge [27]. The motorbikes have more variability in their appearance and the views are quantified finer because of the larger variability in

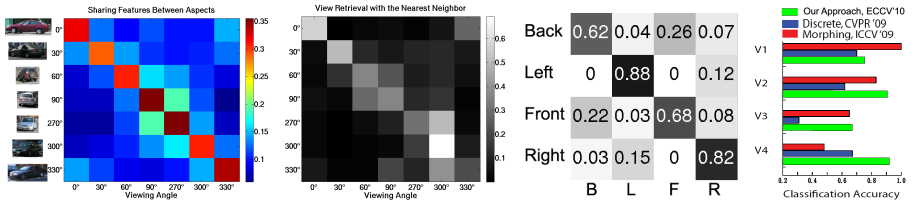


(a) (b) (c)

Fig. 4. (a) Joint voting achieves better results than separate voting because of insufficient training data per view and finely quantized views although it does not use view annotations. Estimating the bounding box from backprojection even leads to a small improvement. (b) Sharing of features across views of motorbikes. (c) Performance comparison to state-of-the-art multi-view detectors for the motorbikes dataset.

aspect ratios. For the sake of a fair comparison the same training and test settings as in [5] is used. The results of this experiment are shown in Fig. 4(a). Note that the detection result with joint voting is obtained only using the bounding box annotations for the training data and using no view annotations. It is important to note that the aim of the experiments is to show improvements over our baselines with the same parameters throughout experiments. The performance of joint voting and other state-of-the-art approaches is shown in Fig. 4(c) to give the reader an idea of the performance of other approaches compared to ours on this dataset. Note that in Thomas et al. [5] and [17] pixel accurate segemented training data is used. In contrast to our approach and [5], the sliding window approach of Savarese et al. [17] explicitly uses the geometrical relationships of different views. Although these relationships seem to be useful (better recall) for detection it comes at high computational costs which makes this approach not scalable to large datasets. In particular, testing with this approach has linear complexity in the number of training examples compared to logarithmic in our implementation. And training complexity is quadratic in the number of training images (linear in our case). In addition, although this work does not use view annotations, unlike our approach it needs many views of several training examples which are expensive to acquire.

Sharing features across views One of the main advantages of training multiple views jointly is the sharing of features. In order to evaluate the capability of our method in doing so, we are creating a sharing matrix of size $n_{views} \times n_{views}$. Each element of this matrix shows, on average, how many features of the column view are used for detection of the row view. Since the test set of none of the datasets is annotated for views, this experiment is done on the set of training data with a leave-one-out strategy. When running the detector on a training instance, we are removing all the occurrences that are originating from that instance from the forest. The sharing matrices for the Leuven-cars and Leuven-motorbikes datasets are shown in Figs. 4(b) and 5(a).

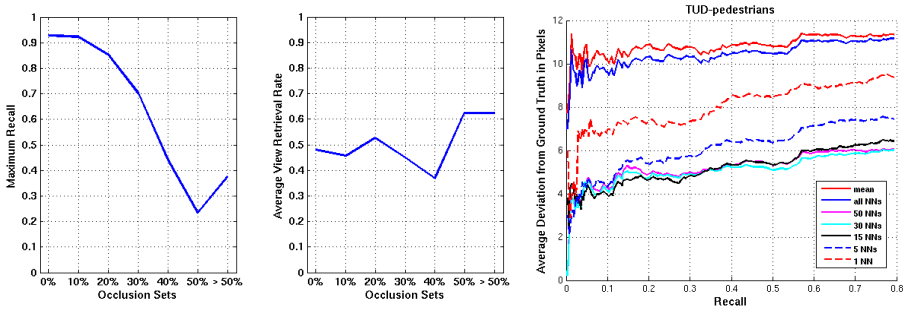


(a) (b) (c) (d)

Fig. 5. (a) Sharing codebook occurrences across views for Leuven-cars. (b) Viewpoint retrieval with the nearest neighbor using (13). (c) View-point classification for detected cars in the VOC'06 dataset. As can be seen, the confusion appears to be limited only to similar views. (d) Comparison to state-of-the-art [16,18] for view classificaiton on VOC'06 (for a fair comparison detections up to 43% recall are considered; average accuracy 82%). Note that our nearest neighboring approach leads to superior performance and more balanced estimation. Comparing the sharing pattern and view confusions is also interesting; e.g. front and back views share many features but their view have been separated well. This shows the presence of additional information in the support.

4.2 Estimating View-point with Nearest Neighbors

As described in Sect. 3.3, support intersection can be used as metric to compare two detections. In order to assess the quality of this metric, we use it to retrieve the viewpoint of the detected cars in the Leuven and PASCAL VOC'06 cars datasets. To this end, we have hand-annotated the viewpoint of the full Leuven-cars test set. For the PASCAL VOC'06 cars set, the ground truth annotations were used. For the Leuven-cars, we have run the detector on the positive set of the training data and collected a set of detections. For the VOC'06 set, the same procedure is carried out but on the validation set. All detections are done with joint voting (see Sect. 3.2) and not using view annotations. By comparing the support of a test detection to the support of all positive collected detections using (13), the nearest neighbor is retrieved and the estimated view of it is assigned to the test detection. This has been done for all the true positives in the test set and their estimated viewpoint is stored. By comparing the estimated viewpoint with the ground truth annotations, the confusion matrix in Fig. 5(b) (with average diagonal of 43%) is created where the rows indicate the ground-truth viewpoints and columns are the estimated viewpoints. In order to see if retrieving more nearest neighbors would add robustness to this process, this experiment is repeated by retrieving the 35 nearest training detections for each test detection and assigning the viewpoint of the majority to it (with average diagonal of 50%). The results for the VOC'06 are given in Fig. 5(c,d). As can be seen, most confusion is happening between very similar views. Note that the features used in our detection system are relatively invariant with respect to small viewpoint changes and the training is done without using viewpoint annotations and in a way to optimize detection performance. In addition, there is a relatively large overlap in the annotation of nearby views due to the difficulty of precise viewpoint estimation even for humans. A video showing the estimated



(a) (b) (c)

Fig. 6. (a-b) The view retrieval performance using (13) together with the proportion of the cars detected depending on the amount of occluded regions for a subset of the Leuven-car sequence. (the last two sets, 50% and $> 50\%$, have very few instances). The recall and view retrieval performances were calculated independently for each occlusion set. Interestingly, although the detection performance deteriorates from large occlusions (a), the viewpoint retrieval performance is affected very little (b) which shows robustness of this similarity measure to occlusions. (c) Distance between the ankles estimated by the median of the k nearest training images using (12) compared to mean and median as baselines. The estimation is robust even at high recall rates.

views for the entire Leuven-cars dataset is available under `http://www.vision.ee.ethz.ch/~nrazavi`.

The effect of occlusion: In order to assess the quality of the support intersection similarity metric in the presence of occlusions, we have annotated all the cars in every tenth frame of the Leuven-cars sequence based on the amount of occlusion: not occluded, 10%, 20%, 30%, 40%, and $> 50\%$ occluded regions. In this experiment, first the detector, with the same settings as in the multi-view experiment, Sect. 4.1, is applied to all the images and a number of detections are retrieved for each image. Then for each correct detection, its viewpoint is estimated as described above. For each occlusion set, we have evaluated how accurately the viewpoint is estimated. The results in Fig. 6(b) show the robustness of this nearest neighbor metric with respect to partial occlusions.

4.3 Retrieving Nearest Training Examples

In Sect. 3.3, we have explained how backprojection can be used as a similarity measure between object hypothesis and the training examples. In the following experiment, we are using such information to estimate the distance between the ankles of pedestrians as an indicator of their pose; see Fig. 7. We carried out our experiments on the TUD-pedestrians dataset. Training data of this dataset has annotations of the joint positions and this information is exploited for estimating the Euclidean distance (in pixels) between the ankles of a test instance. For the sake of evaluation, we have produced the same annotations for the test set. The

Fig. 7. Two test detections from TUD-pedestrians dataset and their top ten nearest training examples (top row; nearest examples ordered from left to right) and backprojections of detection support to them (bottom row) using (12). The blue box shows the estimated bounding box from the backprojection mask (blended). Note the similarity of the poses between the test instances and retrieved nearest training images.

distance between the ankles of the test instance is then estimated as the median of this distance in the k NNs. Figure 6(c) shows the deviation of the estimated distance from the ground truth for different values of k. As a baseline, we also show the deviation from the ground truth if the distance is estimated by the mean or median distance of the whole training set.

5 Conclusions

We have introduced an extension of the Hough-based object detection to handle multiple viewpoints. It builds a shared codebook by considering different viewpoints jointly. Sharing features across views allows for a better use of training data and increases the efficiency of training and detection. The performance improvement of sharing is more substantial with few training data. Moreover, we have shown that the support of a detection and its backprojection can be exploited to estimate the extent of a detection, retrieve nearest training examples, and establish an occlusion-insensitive similarity measure between two detections.

Although the verification of object hypotheses is not the focus of this work, the detection performance is likely to improve by an additional verification step like MDL [3]. Moreover, the backprojection masks could be used in combination with a CRF to obtain object segmentations similar to [28]. The similarity metrics could be used in the context of SVM-KNN [29] for verification.

Acknowledgments: We wish to thank the Swiss National Fund (SNF) for support through the CASTOR project (200021-118106).

References

1. Agarwal, S., Awan, A., Roth, D.: Learning to detect objects in images via a sparse, part-based representation. TPAMI **26** (2004) 1475–1490
2. Ballard, D.H.: Generalizing the hough transform to detect arbitrary shapes. Pattern Recognition **13** (1981) 111–122
3. Leibe, B., Leonardis, A., Schiele, B.: Robust object detection with interleaved categorization and segmentation. IJCV **77** (2008) 259–289
4. Thomas, A., Ferrari, V., Leibe, B., Tuytelaars, T., Van Gool, L.: Using multi-view recognition and meta-data annotation to guide a robot's attention. Int. J. Rob. Res. **28** (2009) 976–998

5. Thomas, A., Ferrari, V., Leibe, B., Tuytelaars, T., Schiele, B., , Gool, L.V.: Towards multi-view object class detection. In: CVPR. (2006)
6. Leibe, B., Cornelis, N., Cornelis, K., Gool, L.V.: Dynamic 3d scene analysis from a moving vehicle. In: CVPR. (2007)
7. Opelt, A., Pinz, A., Zisserman, A.: Learning an alphabet of shape and appearance for multi-class object detection. IJCV (2008)
8. Shotton, J., Blake, A., Cipolla, R.: Multiscale categorical object recognition using contour fragments. TPAMI **30** (2008) 1270–1281
9. Boiman, O., Shechtman, E., Irani, M.: In defense of nearest-neighbor based image classification. In: CVPR. (2008)
10. Gall, J., Lempitsky, V.: Class-specific hough forests for object detection. In: CVPR. (2009)
11. Maji, S., Malik, J.: Object detection using a max-margin hough transform. In: CVPR. (2009)
12. Ommer, B., Malik, J.: Multi-scale object detection by clustering lines. In: ICCV. (2009)
13. Selinger, A., Nelson, R.C.: Appearance-based object recognition using multiple views. CVPR (2001)
14. Seemann, E., Leibe, B., , Schiele, B.: Multi-aspect detection of articulated objects. In: CVPR. (2006)
15. Kushal, A., Schmid, C., Ponce, J.: Flexible object models for category-level 3d object recognition. In: CVPR. (2007)
16. Su, H., Sun, M., Fei-Fei, L., Savarese, S.: Learning a dense multi-view representation for detection, viewpoint classification and synthesis of object categories. In: ICCV. (2009)
17. Savarese, S., Fei-Fei, L.: 3d generic object categorization, localization and pose estimation. In: ICCV. (2007)
18. Sun, M., Su, H., Savarese, S., Fei-Fei, L.: A multi-view probabilistic model for 3d object classes. In: CVPR. (2009)
19. Chiu, H.P., Kaelbling, L., Lozano-Perez, T.: Virtual training for multi-view object class recognition. In: CVPR. (2007)
20. Farhadi, A., Tabrizi, M., Endres, I., Forsyth, D.: A latent model of discriminative aspect. In: ICCV. (2009)
21. Breiman, L.: Random forests. Machine Learning **45** (2001) 5–32
22. Everingham, M., Van Gool, L., Williams, C.K.I., Winn, J., Zisserman, A.: The 2006 pascal visual object classes challenge (2006)
23. Blaschko, M.B., Lampert, C.H.: Learning to localize objects with structured output regression. In: ECCV. (2008)
24. Lazebnik, S., Schmid, C., Ponce, J.: Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In: CVPR. (2006)
25. Andriluka, M., Roth, S., Schiele, B.: People-tracking-by-detection and people-detection-by-tracking. In: CVPR. (2008)
26. Torralba, A., Murphy, K.P., Freeman, W.T.: Sharing visual features for multiclass and multiview object detection. TPAMI **29** (2007) 854–869
27. Everingham, M., et al.: The 2005 pascal visual object classes challenge. (2005)
28. Winn, J.M., Shotton, J.: The layout consistent random field for recognizing and segmenting partially occluded objects. In: CVPR (1). (2006) 37–44
29. Zhang, H., Berg, A.C., Maire, M., Malik, J.: Svm-knn: Discriminative nearest neighbor classification for visual category recognition. In: CVPR. (2006)