Multi-View Priors for Learning Detectors From Sparse Viewpoint Data

Bojan Pepik[1] **Michael Stark**[1,2] **Peter Gehler**[3] **Bernt Schiele**[1]
[1]Max Planck Institute for Informatics, [2]Stanford University,
[3]Max Planck Institute for Intelligent Systems

Abstract

While the majority of today's object class models provide only 2D bounding boxes, far richer output hypotheses are desirable including viewpoint, fine-grained category, and 3D geometry estimate. However, models trained to provide richer output require larger amounts of training data, preferably well covering the relevant aspects such as viewpoint and fine-grained categories. In this paper, we address this issue from the perspective of transfer learning, and design an object class model that explicitly leverages correlations between visual features. Specifically, our model represents prior distributions over permissible multi-view detectors in a parametric way – the priors are learned once from training data of a source object class, and can later be used to facilitate the learning of a detector for a target class. As we show in our experiments, this transfer is not only beneficial for detectors based on basic-level category representations, but also enables the robust learning of detectors that represent classes at finer levels of granularity, where training data is typically even scarcer and more unbalanced. As a result, we report largely improved performance in simultaneous 2D object localization and viewpoint estimation on a recent dataset of challenging street scenes.

1 Introduction

Motivated by higher-level tasks such as scene understanding and object tracking it has been argued that object class models should not only provide flat, 2D bounding box detections but rather provide more expressive output, such as a viewpoint estimate [39, 43, 25, 40, 32, 45, 18] or an estimate of the 3D geometry of the object of interest [34, 47, 33, 14, 29, 49]. Similarly, there has been in increased interest in object representations that allow more fine-grained distinctions than basic-level categories [24, 21, 42], for two reasons. First, these representations potentially perform better in recognition, as they explicitly address the modes of intra-class variation. And second, they, again, can provide further inputs to higher-level reasoning (e.g., in the form of fine-grained category labels).

However, today's methods for 3D and fine-grained object representations suffer from a major weakness: for robust parameter estimation, they tend to require an abundance of annotated training data that covers all relevant aspects (viewpoints, sub-categories) with sufficient density. Unfortunately, this abundance of training data cannot be expected in general. Even in the case of dedicated multi-view datasets [39, 31, 1, 42] or when resorting to artificially rendered CAD data [25, 40, 50], the distribution of the number of available training images over object categories is known to be highly unbalanced and heavy-tailed [46, 38, 27]. This is particularly pronounced for categories at finer levels of granularity, such as individual types or brands of cars[1].

Transfer learning has been acknowledged as a promising way to leverage scarce training data, by reusing once acquired knowledge as a regularizer in novel learning tasks [11, 41, 16]. While it has

[1]Fig. 5 and 6 in Sect. 6 show the viewpoint data statistics for *car-types* and *car-models* on KITTI [17].

been shown that transfer learning can be beneficial for performance, its use in computer vision has, so far, mostly been limited to either classification tasks [11, 51, 37, 5] or flat, 2D bounding box detection [2, 16], neglecting both the 3D nature of the recognition problem and more fine-grained object class representations.

The starting point and major contribution of this paper is therefore to design a transfer learning technique that is particularly tailored towards multi-view recognition (encompassing simultaneous bounding box localization and viewpoint estimation). It boosts detector performance for scarce and unbalanced training data, lending itself to fine-grained object representations.

To that end, we represent transferable knowledge as prior distributions over permissible models [11, 16], in two different flavors. The first flavor (Sect. 3.1) captures sparse correlations between HOG cells in a multi-view deformable part model (DPM [12]), across viewpoints. While this is similar in spirit to [16] in terms of statistical modeling, we explicitly leverage 3D object geometry [34, 33] in order to establish meaningful correspondences between HOG cells, in a fully automatic way. As we show in our experiments (Sect. 4), this already leads to some improvements in performance in comparison to [16]. The second flavor (Sect. 3.2) extends the sparse correlations to a full, dense covariance matrix that potentially relates each HOG cell to every other HOG cell, again across viewpoints – this can be seen as directly learning transformations between different views, where the particular choice of source and target cells can function as a regularizer on the learned transformation, and leads to substantial improvements in simultaneous bounding box localization and viewpoint estimation. Both flavors are simple to implement (covariance computation for prior learning and feature transformation for prior application) and hence widely applicable, yet lead to substantial performance improvements for realistic training data with unbalanced viewpoint distributions.

Our paper makes the following specific contributions: *First*, to our knowledge, our work is the first attempt to explicitly design a transfer learning technique for multi-view recognition and fine-grained object representations. *Second*, we propose two flavors of learning prior distributions over permissible multi-view detectors, one based on sparse correlations between cells and one based on the full covariance. Both are conveniently expressed as instantiations of a class of structured priors that are easy to implement and can be readily applied to current state-of-the-art multi-view detectors [12, 34, 33]. And *third*, we provide an in-depth experimental study of our models, first investigating multi-view transfer learning under the controlled conditions of a standard multi-view benchmark [39] (Sect. 4.1), and finally demonstrating improved performance for simultaneous object localization and viewpoint estimation on realistic street scenes [17] (Sect. 4.2).

2 Related work

The problem of scarce training data has mainly been addressed in two different ways in the literature.

Generating training data. The first way is to explicitly generate more training data for the task at hand, e.g., by rendering CAD models of the object class of interest. Rendered data has successfully been applied in the context of multi-view recognition [26, 25, 40, 50, 34, 33], people detection [36, 35], and indoor scene understanding [7]. The success of these approaches is due to the fact that unlimited amounts of training data can be generated with lots of variation in viewpoint, shape and articulation, from just a few models. Existing approaches differ mostly in the acquisition of appropriate models (3D scanning [36, 35] vs. manual design [26, 25, 40, 50, 34, 33]) and the rendering output, ranging from close to photo-realistic images [35, 26] to directly rendering edge-based features [40, 36, 34, 33]. A special case of data "generation" is the borrowing of training examples from other object classes [27], adapting feature representations to exploit data from different domains [22] or using decorrelated features [20]. While our model based on local correlation structures (Sect. 3.1) can leverage CAD models of the object class, this is done for the sole purpose of deriving correspondences based on 3D geometry, and does not include object appearance.

Transfer learning. The second way is to first condense available training data into a model, and then reusing that model in the context of another learning task. While there is a vast amount of literature on this kind of transfer learning, most approaches focus on object or image classification rather than detection [30, 15, 4, 3, 44, 51, 23, 37, 5]. In terms of detection, approaches range from shape-based object class representations relying on manually-annotated parts in a single view [41]

over HOG- ([6]) templates with fixed layout [2] to full-fledged deformable part models [16] that share low-level feature statistics. Common to all these approaches is that they are agnostic about the inherent 3D nature of the object class detection problem; in contrast, our models explicitly leverage 3D (Sect. 3.1) or viewpoint (Sect. 3.2) information. This is a key difference in particular to [16], which focuses entirely on low-level features that can be universally transferred. Since part of our evaluation is performed on object categories of a fine granularity (individual car-models), we acknowledge that this is of course connected to work in fine-grained categorization [48, 10, 42]. In contrast to these works, however, the focus of ours is on learning multi-view detectors from scarce training data, rather than classifying cropped images according to a fine-grained class taxonomy.

3 Multi-view transfer learning

We consider the scenario of transfer learning for object models. The goal is to train an object detection model for a target class for which only very few labeled examples are available. However we have access to an existing (or several) object model for a similar (or the same) object class, the source models. The main intuition that guides our approach is that if we extract common regularities shared by both object classes, then this in turn can be used to devise better target models. In the case of object detection on HOG [6] we reason that although the actual feature distribution may differ, there are similarities in how the features deform under transformations such as viewpoint changes.

Preliminaries. More formally, let us denote by $\mathbf{w}^s$ the parameters of a source model. Specifically in the case of multi component detectors we have $\mathbf{w}^s = [w_1^s, \ldots, w_C^s]$, where the individual w_i^s denote different components of the models. As we are interested in the multi-view setting, the components represent specific viewpoints in our case. Given $\mathbf{w}^s$ and a few N_t labeled examples of a target class $\{x_i, y_i; i \in \{1, \ldots, N_t\}\}$, the goal is to derive a detection model $\mathbf{w}^t$. This is implemented via the regularized risk functional, which has been used for multi-task learning in [9]

$$\mathbf{w}^t = \operatorname*{argmin}_{\mathbf{w}} J(\mathbf{w}) + \sum_{i=1}^{N_t} l(\mathbf{w}, x_i, y_i), \tag{1}$$

consisting of a regularization $J(\mathbf{w})$ and a data fit term, here the empirical loss l on the training data points. The data term is standard and we may use loss functions such as structured losses or simpler losses like the Hinge loss for classification. In addition to the data term we regularize the model parameters with J, that is derived using information from the source model. We use a transfer learning objective based on [9] where the same regularizer is proposed in the context of multi-task learning. The regularizer is quadratic and of the form

$$J(\mathbf{w}) = \mathbf{w}^\top K_s \mathbf{w}.$$

We distinguish between different choices of K_s, implementing different possibilities to transfer knowledge from the source model. When $K_s = \mathbf{I}$, the objective (1) reduces to the standard **SVM** case. In the following, we will explore three different variants for the knowledge transfer matrix K_s.

3.1 Learning sparse correlation structures

Let us denote with w an appearance filter of one viewpoint component in the entire set of parameters $\mathbf{w}$. For simplicity we will simply refer to this as w without using sub- or superscripts. This filter is of size $n \times m \times L$. It has spatial extent of $n \times m$ cells, and L are filter values computed in each cell ($L = 32$ in [12]). We denote each cell j as a vector $w_j \in \mathbb{R}^L$. We implement different versions of the transfer learning objective (1) using a graph Laplacian regularization approach ([9], Sect 3.1.3) by choosing

$$J(\mathbf{w}) = \mathbf{w}^\top K_s \mathbf{w} = \mathbf{w}^\top (I - \lambda \Sigma_s) \mathbf{w},$$

where Σ_s encodes correlations between different cells in the model. The matrix Σ_s is of size $P \times P$, with P being the total number of model parameters. To distinguish between different choices for the structure of Σ_s we denote with $\sim_n$ a relationship of type n between two cells in w. With "type", for example we can refer to a spatial relation among cells, such as horizontal neighbors, vertical neighbors, etc. This defines a set of cell pairs $\mathbb{P}_n = \{(w_j, w_k) | j \sim_n k\}$ in the model that satisfy relation $\sim_n$. From the set $\mathbb{P}_n$ one can compute cross covariances for different types

$$\Sigma_n = \sum_{j \sim_n k} (w_j - \bar{w})(w_k - \bar{w})^\top, \tag{2}$$

where $\bar{w} = \frac{1}{|\mathbb{P}_n|} \sum_j w_j$ is the mean of the set of cells. The full $P \times P$ matrix Σ_s is then constructed from the smaller $L \times L$ block matrices Σ_n (details below). This results in a sparse Σ_s, as the number of cell pairs satisfying a relation is small compared to the total number of possible cell pairs.

Single view correlation structures (SVM-SV). Originally proposed in [16], **SVM-SV** aims at capturing generic neighborhood relationships between HOG cells within a single template (i.e., a single view). This implements a specific choice for $\sim_n$. **SVM-SV** focuses on 5 specific relation types: horizontal ($\sim_h$), vertical ($\sim_v$), upper-left diagonal ($\sim_{d1}$) and upper-right diagonal ($\sim_{d2}$) cell relations. In addition, **SVM-SV** captures self-correlations of the same cell $\sim_{cell}$. For a given relation type $\sim_n$, **SVM-SV** takes into account all cell pairs in the template which satisfy the relation $\sim_n$, to compute each of the different cross-covariances Σ_h, Σ_v, Σ_{d1}, Σ_{d2} and Σ_{cell}.

Multi-view correlation structures (SVM-MV). We extend **SVM-SV** to encompass multi-view knowledge transfer. In our model different components w of $\mathbf{w}$ correspond to different viewpoints of the object class. Different components are very related since they have a common cause in the geometric structure of the three dimensional object. Therefore, the goal of **SVM-MV** is to capture the across-view cell relations in addition to the single view cell relation introduced by **SVM-SV**. For that purpose, we learn a new, across-view relation type $\sim_{mv}$, capturing cell relationships across different views.

In order to establish cell relationships across viewpoints, we use a 3D CAD model of the object class of interest (or a generic 3D ellipsoid with proper aspect ratio in case we do not have a CAD model available for a class), which provides a unique 3D reference frame across views. The alignment between learned and 3D CAD models is achieved by back-projecting 2D cell positions onto the 3D surface, assuming known viewpoints for the learned models and fixed perspective projection. We then establish cell relationships between cells that back-project onto the same 3D surface patch in neighboring views, and learn Σ_{mv}.

After computing the different cross-covariances Σ_n for both **SVM-SV** and **SVM-MV** from the source models, we construct the Σ_s matrix, which is further on used as a regularizer in the target model training (Eq. 1). Σ_s uses the learned cell-cell correlations of different types from the source models, to guide the training of the target model. In order to construct Σ_s, we first establish pairs of cells (w_i, w_j) in the target model which satisfy a certain relation type $\sim_n$ (e.g. neighbors across views) and then we populate the corresponding entries in Σ_s, $\Sigma_s^{i,j}$ with Σ_n. We apply this procedure for all cell relation types defined in **SVM-SV** and **SVM-MV**.

3.2 Learning dense multi-view correlation structures (SVM-Σ)

SVM-MV and **SVM-SV** capture correlation structures among model cells that satisfy certain cell relations (2D neighboring cells, 3D object surface) resulting in a sparse graph encoded by Σ_s. In the following, we extend this limited structure to a dense graph, that potentially captures relationships among all cells in the model. We will refer to this model as **SVM-**Σ.

Let's assume we are given a set of N source models $\{\mathbf{w}_1^s, \ldots, \mathbf{w}_N^s\}$, for example by training several models using bootstrapping. Then, we compute the un-normalized covariance matrix Σ_{emp}

$$\Sigma_{emp} = \frac{1}{N} \sum_{i=1}^{N} \mathbf{w}_i^s \mathbf{w}_i^{s\top} \tag{3}$$

which is a rank N matrix. We set $\Sigma_s = \Sigma_{emp}$.

This variant of Σ_s (**SVM-**Σ) is dense and captures correlations of all types among all cells across all viewpoints in the model. Unlike **SVM-MV** and **SVM-SV**, which are rather generic in nature (e.g., all pairs of horizontal pairs in the template are considered when learning Σ_h), **SVM-**Σ can capture very specific and local cell correlations, within a single template (view) and across views. Fig. 1 (left) visualizes a heatmap of the strength of the learned correlations for **SVM-**Σ between given reference cells (red squares, black cubes) and all other cells, back-projected onto the 3D surface of a car CAD model. Note that the heatmaps indeed reflect meaningful relationships (e.g., the front wheel surface patch shows high correlation with back wheel patches).

While **SVM-**Σ is a symmetric prior (as the correlations are computed across all views in the model), we also consider the case where the target training data distribution is sparse over viewpoints. We

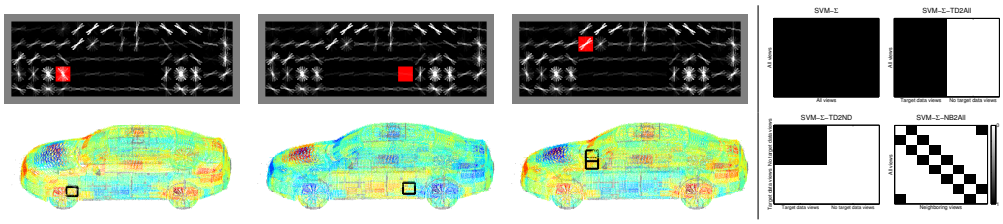


Figure 1: (Left) Learned priors visualized in 3D (for a *reference* cell). Red indicates the *reference* cell. The black cube indicates the *reference* cell back-projected into 3D. (Right) **SVM-Σ** versions.

address this by sparser, asymmetric variants of **SVM-Σ** that connect only certain views with each other, by zeroing out parts of the Σ_s using an element-wise multiplication with a sparse matrix S as $S \circ \Sigma_s$. Several choices of S are depicted in Figure 1 (right). We distinguish between the following asymmetric priors: **SVM-Σ-TD2ND**, where we transfer knowledge from views for which we have target data to views with no target training data, **SVM-Σ-TD2ALL** with transfer from views with target data to all views, and **SVM-Σ-NB2ALL** where we transfer from every viewpoint to its neighboring viewpoints.

3.3 Learning a target model using the learned K_s matrix

We perform model learning (Eq. 1) by first doing a Cholesky decomposition of $K_s = U^\top U$. This allows us to define feature and model transformations: $\tilde{\mathbf{x}} = U^{-\top}\mathbf{x}$ and $\tilde{\mathbf{w}} = U\mathbf{w}$. Using these transformations, one can show that $\mathbf{w}^\top K_s \mathbf{w} = \tilde{\mathbf{w}}^\top \tilde{\mathbf{w}}$ and $\tilde{\mathbf{w}}^\top \tilde{\mathbf{x}} = \mathbf{w}^\top \mathbf{x}$, which means we can learn a target model by first, transforming the features and the models using U, then training a model via a standard **SVM** solver in the transformed space, and in the end transforming back the trained model. In the **SVM-SV** case, to be compatible with [16], we perform eigen decomposition instead of Cholesky.

4 Experiments

In this section, we carefully evaluate the performance of our multi-view priors. First (Sect. 4.1), we provide an in-depth analysis of different variants of the **SVM-MV** and **SVM-Σ** priors in a controlled training data setting, by varying the viewpoint distribution of the training set. We learn target models using a few target training examples plus our priors and compare them to using the **SVM-SV** prior proposed in [16] and using standard **SVM**. We perform the analysis on two tasks, 2D bounding box localization and viewpoint estimation on the 3D Object Classes dataset [39], demonstrating successful knowledge transfer even for cases in which there is no training data for $3/4$ of the viewing circle. Second (Sect. 4.2), we highlight the potential of our **SVM-Σ** priors to greatly improve the performance of simultaneous 2D bounding box localization and viewpoint estimation in a realistic, uncontrolled data set of challenging street scenes (the tracking benchmark of KITTI [17]).

For computational reasons, we restrict ourselves to the root-template-only version of the DPM [13] as the basis for all our models in Sect. 4.1, but consider the full, part-based version for the more challenging and realistic experiments in Sect. 4.2. In all cases, the C parameter is fixed to 0.002 [12] for all tested methods. We set $\lambda = 0.9/e_{max}$, where e_{max} is the biggest eigenvalue of Σ_s. We empirically verified that this always resulted in a positive definite matrix K_s, which makes Eq. 1 a convex optimization problem.

4.1 Comparison of multi-view priors

We start by comparing the different multi-view priors on the 3D Object Classes dataset [39] (a widely accepted multiview-benchmark with balanced training and test data from 8 viewpoint bins, for 9 object classes), in two sets of experiments. In the first set, we use the same number k of target training examples per view (multi-view *k-shot* learning). In the second set, we exclude certain viewpoints completely from the training data ($k = 0$), keeping only a single example from each of the other viewpoints (sparse multi-view *k-shot* learning). In both cases, the test set requires detecting

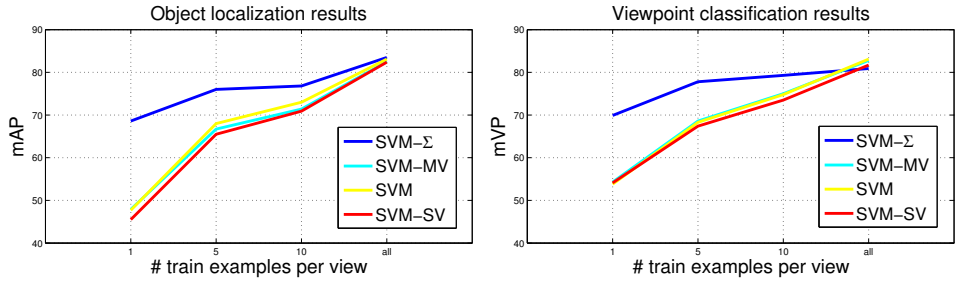


Figure 2: 2D BB localization (left) and viewpoint estimation (right) on 3D Object Classes [39].

	SVM-SV	**SVM-MV**	**SVM-Σ**	**SVM**	[28]	[19]	DPM-hinge+VP [34]	DPM-VOC+VP [34]	[25]	[50]	[32]	[18]
car	99.6 / 92.9	99.8 / 95.0	99.8 / 92.5	99.8 / 95.0	96.0 / 89.0	-/-	99.6 / 92.5	99.8 / 97.5	76.7/70.0	90.4/84.0	-/86.1	99.2/85.3
bicycle	88.8 / 87.6	89.9 / 87.6	96.7 / 92.2	90.1 / 87.9	91.0 / 88.0	-/-	98.6 / 93.0	98.8 / 97.5	69.8 / 75.5	-/-	-/80.8	-/-
iron	94.9 / 94.7	96.4 / 96.1	90.6 / 88.8	97.0 / 95.5	53.0 / -	-/-	93.3 / 86.3	96.0 / 89.7	-/-	-/-	-/-	-/-
cell.	51.0 / 82.2	51.2 / 82.3	53.7 / 80.9	51.1 / 81.5	43.0 / -	-/-	62.9 / 65.4	62.4 / 83.0	-/-	-/-	-/-	-/-
mouse	61.3 / 74.7	61.2 / 70.8	61.4 / 69.8	62.5 / 72.5	41.0 / -	-/-	73.1 / 62.2	72.7 / 76.3	-/-	-/-	-/-	-/-
shoe	93.9 / 81.4	93.4 / 87.3	94.7 / 86.2	94.8 / 86.5	78.0 / -	-/-	97.9 / 71.0	96.9 / 89.8	-/-	-/-	-/-	-/-
stapler	71.5 / 69.2	72.6 / 70.2	74.2 / 70.2	74.2 / 70.2	32.0 / -	-/-	84.4 / 62.8	83.7 / 81.2	-/-	-/-	-/-	-/-
toaster	94.4 / 70.6	95.3 / 72.8	97.2 / 66.7	95.2 / 75.6	54.0 / -	-/-	96.0 / 50.0	97.8 / 79.7	-/-	-/-	-/-	-/-
mAP	81.9 / 81.7	82.5 / 82.8	83.5 / 80.9	83.1 / 83.1	61.0 / 79.2	- / 74.2	88.2 / 72.9	88.5 / 86.8	-/-	-/-	-/-	-/-

Table 1: Comparison to state-of-the-art on 3D Object Classes [39].

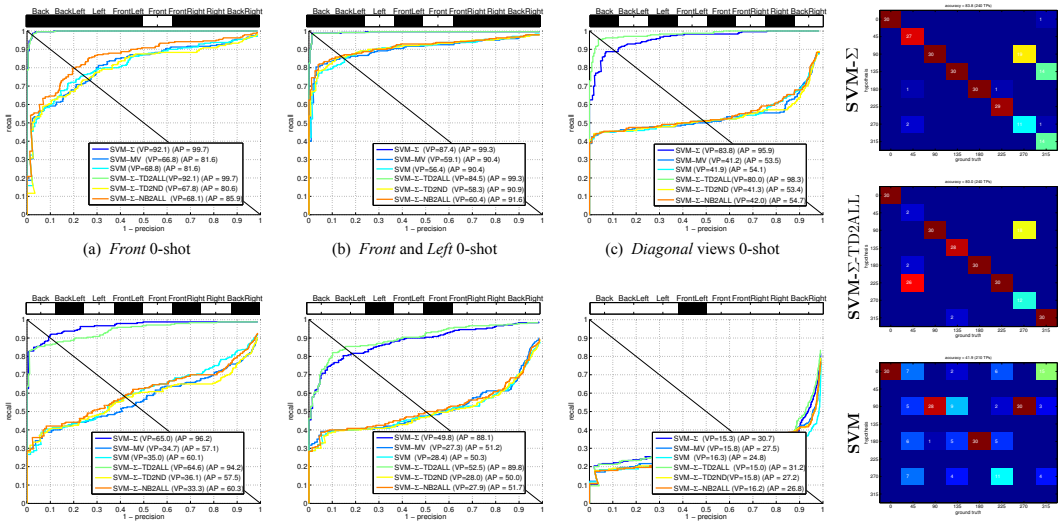
objects seen *from the entire viewing circle*. For each class, our priors are trained using bootstrapping, from 5 source models (each trained from 15 randomly sampled examples per view). The final target model for a class is obtained by using k training examples from that class plus the respective prior.

Multi-view *k-shot* learning. Fig. 2 plots 2D BB localization (left) and viewpoint estimation (right) performance for **SVM**, **SVM-SV**, **SVM-MV**, and **SVM-**Σ, varying the number $k \in \{1, 5, 10, all\}$ of target training examples per view, averaged over 5 randomized runs. We make the following observations. First, we see that **SVM-**Σ outperforms all other methods by significant margins for restricted training data ($k \in \{1, 5, 10\}$), for both 2D BB localization (by at least 20.1%, 8.0% and 3.8%, respectively) and viewpoint estimation (by 15.6%, 9.2% and 4.3%). Second, the benefit of **SVM-**Σ increases with decreasing number of training examples, saturating for $k = all$. And third, **SVM-SV** [16] and **SVM-MV** apparently fail to convey viewpoint-related information beyond what can be learned from the k target examples alone, performing on par with **SVM**.

As a sanity check, Tab. 1 relates the complete per-class results for all methods and $k = all$ (rightmost curve points in Fig. 2) to the state-of-the-art. Despite not using parts, our models in fact outperform previously reported results [19, 25, 50, 28, 32, 18] with and without priors, except for [34] based on DPM [12] with parts. As parts obviously improve performance, we add them in Sect. 4.2.

Sparse multi-view *k-shot* learning. We move on to a more challenging setting in which (single) training examples are only available for selected views, but not for others. Successful localization and viewpoint estimation thus depends on prior information that can be "filled in" for the missing viewpoints. Fig. 3 plots precision-recall curves for the car class and six different settings of increasing difficulty, not having training data for just one view (front) (a), not for two views (front and left) (b), not for four views (diagonal) (c), off-diagonal (d), not for six views (diagonal, back and right) (e), and not for all views except front-left (f). Average precision and viewpoint estimation results are given in plot legends. We compare the performance of **SVM**, **SVM-MV**, **SVM-**Σ, and three further variations of **SVM-**Σ that restrict the structure of the prior covariance matrix (Sect. 3.2), namely, **SVM-**Σ-TD2ALL, **SVM-**Σ-TD2ND, and **SVM-**Σ-NB2ALL.

In Fig. 3 (a) to (f), we observe that two methods succeed in transferring information to up to 6 unseen viewpoints (**SVM-**Σ, dark blue, and **SVM-**Σ-TD2ALL, green), with APs ranging from an impressive 99.7 to 88.1% and VPs ranging from 92.1 to 49.8% for **SVM-**Σ). This observation is confirmed by the confusion matrices in Fig. 3 (g): both **SVM-**Σ and **SVM-**Σ-TD2ALL exhibit a much stronger diagonal structure than **SVM**. Understandably, performance deteriorates for just one observable viewpoint (Fig. 3 (f); AP drops to 30.7%, VP to 15.3%). **SVM-MV** (light blue) provides



(a) *Front* 0-shot (b) *Front* and *Left* 0-shot (c) *Diagonal* views 0-shot

(d) *Off-diagonal* views 0-shot (e) Diagonal, *Back* and *Right* 0-shot (f) *All* views except *FrontLeft* 0-shot (g) Viewpoint confusion matrices

Figure 3: 3D Object Classes [39]. Unbalanced multi-view *0-shot* experiments (on *cars*) with no training data for (a) *Front*, (b) *Front* and Left, (c) *Off-diagonal* views, (d) *Diagonal* views, (e) 1 training example for Left and *Front* views, (f) 1 example for *Front-left* view. (g) VP confusion matrices for the 0-shot *Diagonal* case. Bars on top indicate (with black) which viewpoints are used in training for each experiment.

an advantage over **SVM** (cyan) only for extremely little data (Fig. 3 (e), (f)), where it improves AP by 0.9% and 2.7%.

Summary. We conclude that different kinds of priors (**SVM-SV**, **SVM-MV**, and variations of **SVM-**Σ) vary drastically in their ability to convey viewpoint-related information. Notably, we observe only minor differences between **SVM-SV**, **SVM-MV**, and **SVM**, but large gains in both 2D bounding box localization and viewpoint estimation for **SVM-**Σ.

4.2 Leveraging multi-view priors for object detection

Having verified the ability of our **SVM-**Σ priors to transfer viewpoint information for scarce and unbalanced training data in Sect. 4.1, we now move on to an actual, realistic setting, which naturally exhibits the dataset statistics that we simulated earlier (see Fig. 4, 5 and 6). Specifically, we focus on the *car* object class on the KITTI street scene dataset [17] (and the tracking benchmark subset), consisting of 21 sequences (8,008 images, corresponding to 579 car tracks and $27,300$ annotated car bounding boxes) taken from a driving vehicle. We use 5 sequences for training and the rest for testing. Due to the car-mounted camera setup, the distribution of viewpoints for car objects is already heavily skewed towards back and diagonal views (cars driving in front of the camera car or being parked on the side of the road, see Fig. 4). This becomes even more severe when considering more fine-grained categories, such as individual *car-types* (we distinguish and annotate 7: stat. wagon, convertible, coupe, hatchback, minibus, sedan, suv) and *car-models* (23 in total) [2].

Evaluation criteria. Average precision (AP) computed using the Pascal VOC [8] overlap criterion, based solely on bounding boxes (BB) has been widely used as an evaluation measure. Since the ultimate goal of our approach is to enable simultaneous object localization and viewpoint estimation (both are equally important in an autonomous driving scenario), and in line with [17], we report performance for two combined measures (jointly addressing both tasks) in addition to AP and VP. Specifically, AP+VP-D allows a detection $\hat{y}$ to be a true positive detection if and only if the viewpoint estimate $\hat{y}^v$ is the same as the ground truth y^v. The second measure, AP+VP-C assigns a weight $\hat{w} = (180^\circ - |\angle(\hat{y}^v, y^v)|)/180^\circ$ to the true positive detection based on how well it aligns with the ground truth viewpoint. Also in line with [17], we report results for non-occluded objects.

[2]The annotations will be made publicly available upon publication.

prior		AP / VP			AP+VP-D / AP+VP-C		
method	dataset	*base*	*car-type*	*car-model*	*base*	*car-type*	*car-model*
SVM (KITTI+ 3D obj.)		87.1 / 69.3	- / -	- / -	53.6 / 67.0	- / -	- / -
SVM (KITTI)		86.6 / 68.7	88.7 / 70.9	83.3 / 62.0	53.3 / 65.8	58.1 / 67.9	40.3 / 55.1
SVM-Σ	3D objects	90.7 / 71.9	**91.6** / 75.1	87.5 / 73.9	61.5 / 70.1	65.2 / **74.1**	60.9 / 70.7
SVM-Σ	KITTI	90.7 / 71.9	90.1 / 75.1	89.4 / **75.6**	61.6 / 70.2	**66.1** / 73.5	65.2 / 73.4
SVM-MV	3D objects	90.2 / 72.6	90.3 / 71.9	82.9 / 63.2	60.9 / 69.9	60.8 / 70.0	41.5 / 55.8
SVM-MV	KITTI	89.2 / 73.1	88.5 / 71.1	76.5 / 66.5	62.1 / 69.8	58.8 / 67.6	44.8 / 53.5
SVM-SV	3D objects	90.7 / 71.9	86.5 / 70.4	76.6 / 65.8	61.5 / 70.1	55.9 / 65.8	44.3 / 53.1
SVM-SV	KITTI	86.9 / 71.4	85.8 / 70.8	76.5 / 66.5	59.6 / 67.0	56.5 / 65.1	44.8 / 53.5

Table 2: Multi-view detection results on KITTI [17].

Basic-level category transfer. We commence by applying our priors to a standard object class detector setup, in which a detector is trained such that positive examples are annotated on the level of basic-level categories (i.e., *car*), denoted *base* in the following. Tab. 2 (left) gives the corresponding 2D bounding box localization and viewpoint estimation results, comparing our priors **SVM-MV** and **SVM-**Σ to **SVM-SV** and a baseline not using any prior (**SVM**). For each, we consider two variants depending from which data the prior (or the detector itself for **SVM**) has been trained (KITTI, 3D Object Classes, or both). Note that the respective prior and **SVM** variants use the exact same training data (but in different ways) and are hence directly comparable in terms of performance.

In Tab. 2 (left, col. *base*), we observe that our priors **SVM-MV** and **SVM-**Σ consistently outperform **SVM**, for both 2D BB localization and viewpoint estimation, for both choices of training data (e.g., **SVM-**Σ-KITTI with 90.7% AP and 71.9% VP vs. **SVM**-KITTI with 86.6% AP and 68.7% VP). The performance difference is even more pronounced when considering the combined performance measures (Tab. 2 (right, col. *base*)). **SVM-**Σ-KITTI achieves 61.6% AP+VP-D and 70.2% AP+VP-C, outperforming **SVM**-KITTI (53.3%, 65.8%) by a significant margin.

Similarly, **SVM-**Σ-3D Object Classes outperforms **SVM**-KITTI+3D Object Classes in all measures (90.7% vs. 87.1% AP, 71.9% vs. 69.3% VP, 61.5% vs. 53.6% AP+VP-D and 70.1 vs. 67.0% AP+VP-C). **SVM-MV** and **SVM-SV** priors also show promising detection performance, outperforming the **SVM** models in all metrics.

Fine-grained category transfer. Recently, is has been shown that fine-grained object class representations on the level of sub-categories can improve performance [24, 21, 42], since they better capture the different modes of intra-class variation than representations that equalize training examples on the level of basic-level categories. Further, these representations lend themselves to generate additional output in the form of fine-grained category labels that can be useful for higher-level tasks, such as scene understanding. In the following, we hence consider two fine-grained object class representations that decompose *cars* into distinct *car-types* or even individual *car-models*. Both are implemented as a bank of multi-view detectors (one per fine-grained category) that are trained independently, but combined at test time by a joint non-maxima suppression to yield basic-level category detections.

Note that the individual fine-grained detectors suffer even more severely from scarce and unbalanced training data (see Fig. 5 and 6 in Sect. 6) than on the basic-level (see Fig. 4) – this is where our priors come into play: we train the priors, as before, on the *base* level, and use them to facilitate the learning of each individual fine-grained detector, effectively transferring knowledge from *base* to fine-grained categories.

Tab. 2 gives the corresponding results in columns *car-type* and *car-model*, respectively. We observe: first, performance can in fact improve as a result of the more fine-grained representation, for both **SVM-MV**, **SVM-**Σ and even **SVM** (**SVM**-KITTI-car-type improves AP from 86.6% to 88.7%, and VP from 68.7% to 70.9%, and AP+VP-D from 53.3% to 58.1% and AP+VP-C from 65.8% to 67.9% compared to **SVM**-KITTI-base). A similar boost in performance in viewpoint estimation and combined can be seen for **SVM-**Σ (**SVM-**Σ-KITTI-car-type improves VP from 71.9% to 75.1%, and AP+VP-D from 61.6% to 66.1% and AP+VP-C from 70.2% to 73.5% compared to **SVM-**Σ-KITTI-base; the AP stays consistently high with 90.7% vs. 90.1%). Second, the level of granularity can be too fine: for almost all methods, the performance of the fine-grained *car-model* drops below the performance of the corresponding *base* detector – there is just so little training data for each of the car models that reliable fine-grained detectors can hardly be learned. Curiously, **SVM-**Σ-KITTI-car-model can still keep up in terms of localization (89.4% AP) and even obtains the overall best

		@50 iou				@70 iou			
		AP / VP		AP+VP-D / AP+VP-C		AP / VP		AP+VP-D / AP+VP-C	
prior	dataset	*base*	*car-type*	*base*	*car-type*	*base*	*car-type*	*base*	*car-type*
SVM (KITTI)		90.9 / 74.3	93.2 / 75.9	65.2 / 72.1	66.8 / 74.9	49.9 / 74.2	60.0 / 76.8	37.5 / 40.4	44.6 / 48.3
SVM-Σ	3D obj.	**94.8** / 78.6	93.4 / **81.7**	72.1 / 78.7	**73.0 / 80.6**	51.5 / 81.2	**64.7 / 83.9**	41.9 / 44.3	**53.0 / 56.7**
SVM-Σ	KITTI	94.8 / 77.2	94.3 / 78.3	70.4 / 77.3	70.4 / 79.6	49.7 / 79.0	61.2 / 80.4	39.5 / 41.8	47.9 / 53.3

Table 3: Multi-view detection results on KITTI [17]. Models have root and 4 parts per view.

	without parts							with parts		
prior	**SVM-**Σ	**SVM-**Σ	**SVM-MV**	**SVM-MV**	**SVM-SV**	**SVM-SV**	**SVM**	**SVM-**Σ	**SVM-**Σ	**SVM**
	KITTI	3D obj.	KITTI	3D obj.	KITTI	3D obj.	-	KITTI	3D obj.	-
station wagon	**71.2**	70.2	64.5	63.6	62.6	61.9	61.9	**82.7**	81.9	79.0
convertible	**24.4**	24.0	12.9	10.8	13.8	11.7	12.7	**50.7**	36.8	12.0
coupe	**67.5**	67.1	63.7	67.0	60.5	57.7	67.1	**79.9**	76.6	76.5
hatchback	**89.8**	85.7	66.4	78.2	58.9	65.0	71.0	**95.5**	88.0	87.2
minibus	**31.3**	16.8	20.0	18.7	16.3	18.0	18.6	**59.7**	42.0	41.4
sedan	**69.4**	53.8	46.7	49.4	37.8	41.8	48.7	**83.8**	79.8	66.2
suv	**19.7**	14.7	8.1	7.3	5.2	8.0	8.6	34.5	**35.1**	16.4
mAP	**53.3**	47.5	40.3	42.1	36.4	37.7	41.2	**69.5**	62.9	54.1

Table 4: *Car-type* detection results on the KITTI [17] dataset.

VP accuracy of 75.6%, which is also reflected in the combined measures (65.2% AP+VP-D, 73.4% AP+VP-C). Third, **SVM-**Σ-KITTI-car-type is the overall best method, outperforming the original baseline **SVM**-KITTI-base by impressive margins, in particular for the combined measures (90.1% vs. 86.6% AP, 75.1% vs. 68.7% VP, 66.1% vs. 53.3% AP+VP-D, 73.5% vs. 65.8% AP+VP-C).

Tab. 3 (left) gives the results for the best performing priors of Tab. 2 (**SVM-**Σ-KITTI, **SVM-**Σ-3D Object Classes) in comparison to **SVM**-KITTI, now using parts. As expected, parts result in a general performance boost for all methods (around 5% for all measures). The benefit of our priors remains, for both granularity levels *base* and *car-type*, in particular for the combined measures: **SVM-**Σ-KITTI-base outperforms **SVM**-KITTI-base by similarly large margins as for the no-parts case (70.4% vs. 65.2% AP+VP-D, 77.3% vs. 72.1% AP+VP-C), and **SVM-**Σ-KITTI-car-type outperforms **SVM**-KITTI-car-types by (70.4% vs. 66.8% AP+VP-D, 79.6% vs. 74.9% AP+VP-C).

Tab. 3 (right) applies a tighter overlap criterion for true positive detections (0.7 intersection over union) [17]. Interestingly, this leads to a larger separation in performance between *base* and *car-type* models, in particular in AP: e.g., **SVM-**Σ-3D Object Classes improves from 51.5% to 64.7%, **SVM-**Σ-KITTI from 49.7% to 61.2% and **SVM**-KITTI improves from 49.9% to 60.0%, highlighting the benefit of the fine-grained object class representation in particular for highly precise detection.

Lastly, we evaluate the performance of our fine-grained detectors on the level of the respective fine-grained categories (*car-types*), as independent detection tasks. Tab. 4 gives the results without (left) and with parts (right). Again, our priors **SVM-**Σ consistently outperform the baseline **SVM** for all individual categories as well as on average by large margins (53.3% vs. 41.2% mAP for **SVM-**Σ-KITTI without parts, and 69.5% vs. 54.1% with parts).

Summary. We conclude that our priors (in particular **SVM-**Σ) in fact improve performance for simultaneous 2D bounding box localization and viewpoint estimation, for different levels of granularity of the underlying object representation (*base*, *car-type*, *car-model*). Notably, our priors allow for robust learning even on the most fine-grained level of *car-models*, where training data is scarce and unbalanced and **SVM** fails. The combination of fine-grained representation and prior results in a pronounced performance gain compared to **SVM** on the *base* level.

5 Conclusion

In this paper, we have approached the problem of scarce and unbalanced training data for training multi-view and fine-grained detectors from a transfer learning perspective, introducing two flavors of learning prior distributions over permissible detectors, one based on sparse feature correlations, and one based on the full covariance matrix between all features. In both cases, we have demonstrated improved simultaneous 2D bounding box localization and viewpoint estimation performance when applying these priors to detectors based on basic-level category representations. In addition, the sec-

ond flavor allowed us to learn reliable detectors even for finer-grained object class representations, resulting in an additional boost in performance on a realistic dataset of street scenes [17].

References

[1] M. Arie-Nachimson and R. Basri. Constructing implicit 3D shape models for pose estimation. In *ICCV'09*.

[2] Y. Aytar and A. Zisserman. Tabula rasa: Model transfer for object category detection. In *ICCV'11*.

[3] E. Bart and S. Ullman. Cross-generalization: Learning novel classes from a single example by feature replacement. In *CVPR'05*.

[4] E. Bart and S. Ullman. Single-example learning of novel classes using representation by similarity. In *BMVC'05*.

[5] T. L. Berg, A. C. Berg, and J. Shih. Automatic attribute discovery and characterization from noisy web data. In *ECCV'10*.

[6] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *CVPR'05*.

[7] L. Del Pero, J. Bowdish, B. Kermgard, E. Hartley, and K. Barnard. Understanding Bayesian rooms using composite 3d object models. In *CVPR'13*.

[8] M. Everingham, A. Zisserman, C. K. I. Williams, and L. Van Gool. The PASCAL Visual Object Classes Challenge 2006 (VOC2006) Results, 2006.

[9] T. Evgeniou, C. A. Micchelli, and M. Pontil. Learning multiple tasks with kernel methods. *JMLR'05*.

[10] R. Farrell, O. Oza, N. Zhang, V. I. Morariu, T. Darrell, and L. S. Davis. Birdlets: Subordinate categorization using volumetric primitives and pose-normalized appearance. In *ICCV'11*.

[11] L. Fei-Fei, R. Fergus, and P. Perona. One-shot learning of object categories. *PAMI'06*.

[12] P. F. Felzenszwalb, R. Girshick, D. McAllester, and D. Ramanan. Object detection with discriminatively trained part based models. *PAMI'10*.

[13] P. F. Felzenszwalb, R. B. Girshick, and D. McAllester. Discriminatively trained deformable part models, release 4. http://people.cs.uchicago.edu/ pff/latent-release4/.

[14] S. Fidler, S. Dickinson, and R. Urtasun. 3d object detection and viewpoint estimation with a deformable 3d cuboid model. In *NIPS'12*.

[15] M. Fink. Object classification from a single example utilizing class relevance pseudo-metrics. In *NIPS'04*.

[16] T. Gao, M. Stark, and D. Koller. What makes a good detector? - structured priors for learning from few examples. In *ECCV'12*.

[17] A. Geiger, P. Lenz, and R. Urtasun. Are we ready for autonomous driving? The KITTI vision benchmark suite. In *CVPR'12*.

[18] D. Glasner, M. Galun, S. Alpert, R. Basri, and G. Shakhnarovich. Viewpoint-aware object detection and pose estimation. In *ICCV'11*.

[19] C. Gu and X. Ren. Discriminative mixture-of-templates for viewpoint classification. In *ECCV'10*.

[20] B. Hariharan, J. Malik, and D. Ramanan. Discriminative decorrelation for clustering and classification. In *ECCV*, 2012.

[21] M. Hoai and A. Zisserman. Discriminative sub-categorization. In *CVPR'13*.

[22] B. Kulis, K. Saenko, and T. Darrell. What you saw is not what you get: Domain adaptation using asymmetric kernel transforms. In *CVPR'11*.

[23] C. H. Lampert, H. Nickisch, and S. Harmeling. Learning to detect unseen object classes by between-class attribute transfer. In *CVPR'09*.

[24] T. Lan, M. Raptis, L. Sigal, and G. Mori. From subcategories to visual composites: A multi-level framework for object detection. In *ICCV'13*.

[25] J. Liebelt and C. Schmid. Multi-view object class detection with a 3D geometric model. In *CVPR'10*.

[26] J. Liebelt, C. Schmid, and K. Schertler. Viewpoint-independent object class detection using 3D feature maps. In *CVPR'08*.

[27] J. J. Lim, R. Salakhutdinov, and A. Torralba. Transfer learning by borrowing examples for multiclass object detection. In *NIPS'11*.

[28] R. J. Lopez-Sastre, T. Tuytelaars, and S. Savarese. Deformable part models revisited: A performance evaluation for object category pose estimation. In *ICCV-WS CORP'11*.

[29] M.Hejrati and D.Ramanan. Analyzing 3d objects in cluttered images. In *NIPS'12*.

[30] E. Miller, N. Matsakis, and P. Viola. Learning from One Example Through Shared Densities on Transforms. In *CVPR'00*.

[31] M. Ozuysal, V. Lepetit, and P. Fua. Pose estimation for category specific multiview object localization. In *CVPR'09*.

[32] N. Payet and S. Todorovic. From contours to 3d object detection and pose estimation. In *ICCV'11*.

[33] B. Pepik, P. Gehler, M. Stark, and B. Schiele. 3DDPM - 3d deformable part models. In *ECCV'12*.

[34] B. Pepik, M. Stark, P. Gehler, and B. Schiele. Teaching 3d geometry to deformable part models. In *CVPR'12*.

[35] L. Pishchulin, A. Jain, M. Andriluka, T. Thormaehlen, and B. Schiele. Articulated people detection and pose estimation: Reshaping the future. In *CVPR'12*.

[36] L. Pishchulin, A. Jain, C. Wojek, M. Andriluka, T. Thormaehlen, and B. Schiele. Learning people detection models from few training samples. In *CVPR'11*.

[37] M. Rohrbach, M. Stark, G. Szarvas, I. Gurevych, and B. Schiele. What helps where – and why? semantic relatedness for knowledge transfer. In *CVPR'10*.

[38] R. Salakhutdinov, A. Torralba, and J. B. Tennenbaum. Learning to share visual appearance for multiclass object detection. In *CVPR'11*.

[39] S. Savarese and L. Fei-Fei. 3D generic object categorization, localization and pose estimation. In *ICCV'07*.

[40] M. Stark, M. Goesele, and B. Schiele. Back to the future: Learning shape models from 3d cad data. In *BMVC'10*.

[41] M. Stark, M. Goesele, and B. Schiele. A shape-based object class model for knowledge transfer. In *ICCV'09*.

[42] M. Stark, J. Krause, B. Pepik, D. Meger, J. Little, B. Schiele, and D. Koller. Fine-grained categorization for 3d scene understanding. In *BMVC'12*.

[43] H. Su, M. Sun, L. Fei-Fei, and S. Savarese. Learning a dense multi-view representation for detection, viewpoint classification and synthesis of object categories. In *ICCV'09*.

[44] S. Thrun. Is learning the n-th thing any easier than learning the first. In *NIPS'06*.

[45] M. Villamizar, H. Grabner, J. Andrade-Cetto, A. Sanfeliu, L. V. Gool, and F. Moreno-Noguer. Efficient 3d object detection using multiple pose-specific classifiers. In *BMVC'11*.

[46] J. Wang, J. Yang, K. Yu, F. Lv, T. Huang, and Y. Gong. Locality-constrained linear coding for image classification. In *CVPR'10*.

[47] Y. Xiang and S. Savarese. Estimating the aspect layout of object categories. In *CVPR'12*.

[48] B. Yao, A. Khosla, and L. Fei-Fei. Combining randomization and discrimination for fine-grained image categorization. In *CVPR'11*.

[49] M. Z. Zia, M. Stark, and K. Schindler. Explicit occlusion modeling for 3d object class representations. In *CVPR'13*.

[50] M. Z. Zia, M. Stark, K. Schindler, and B. Schiele. Revisiting 3d geometric models for accurate object shape and pose. In *3dRR-11*.

[51] A. Zweig and D. Weinshall. Exploiting object hierarchy: Combining models from different category levels. In *ICCV'07*.

6 Supplemental material

6.1 Training and test data distributions for *car*, *car-type* and *car-model* category levels

In this section, we visualize the viewpoint distributions of the realistic street scene dataset (the tracking benchmark of KITTI [17]) that is the basis of our experiments in Sect. 4.2.To that end, we plot histograms of the number of car instances for each of 8 viewpoint (azimuth angle) bins, separately for training (blue) and test (red) data, and distinguishing between three different levels of granularity (*car*, *car-type* and *car-model*).

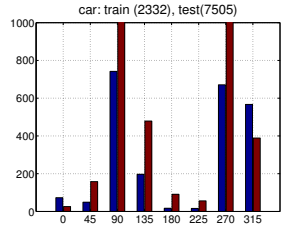


Figure 4: *Car* train and test statistics over 8 viewpoint bins.

Fig. 4 shows the data distribution for the *car* class. Notice the unbalanced data distribution across views. There are two main modes in the viewpoint distribution, at 90° and 270°, which represent back and front views. This behavior is expected, as in the KITTI dataset [17] the images have been taken from a driving vehicle. On the other hand, the left (0°) and right (180°) facing views are poorly represented. With only 16 right facing training examples, training a robust right-view *car* template is extremely challenging.

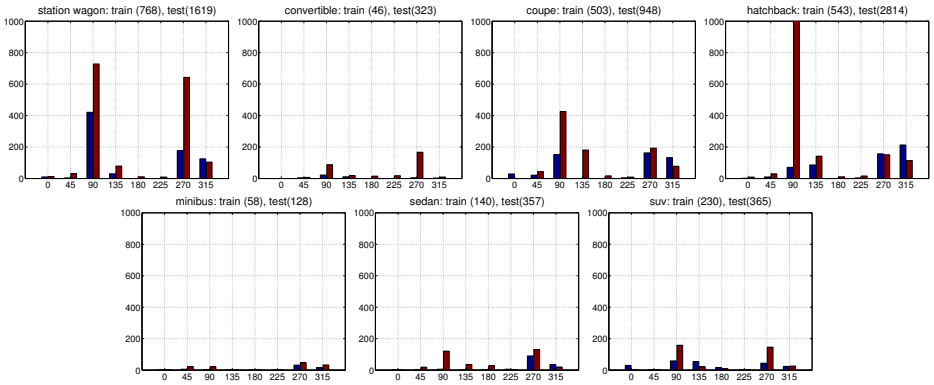


Figure 5: *Car-types* train and test statistics over 8 viewpoint bins.

Fig. 5 illustrates the train and test distributions for each of the 7 *car types* (station wagon, convertible, coupe, hatchback, minibus, sedan, suv). We observe: i) the amount of training data available per *car-type* varies dramatically; classes like *station wagon, coupe, hatchback* have much more examples than *convertible, minibus, sedan*, ii) the training data distributions are skewed. There is not a single *car-type* for which all viewpoints are represented in the training data, thus learning a full multi-view representation for a *car-type* is impossible with a standard **SVM** framework. And iii), the train and test distributions differ a lot for each *car-type*. For example, for *sedan* and *convertible* there are viewpoints in the test set which are not represented in the training set.

Lastly, Fig. 6 illustrates the train and test distribution for each of the 23 different *car-models* we have annotated. For most of the *car-models*, the viewpoint train and test distributions differ drastically. For *car-models* like *VW Touran, VW Passat, Opel Corsa, Opel Tigra, Opel Astra, Ford Ka, Ford Fiesta, Fiat Punto, Fiat Panda, BMW 1* there is either none or at most 1 viewpoint that has both training and test data. This poses a major challenge for learning a robust multi-view object detector. In addition, the absolute number of available training examples for most of the *car-models* is rather small (12 for *Fiat Panda*, 19 for *Mini Cooper*, 12 for *Opel Astra* etc.). In summary, the training and

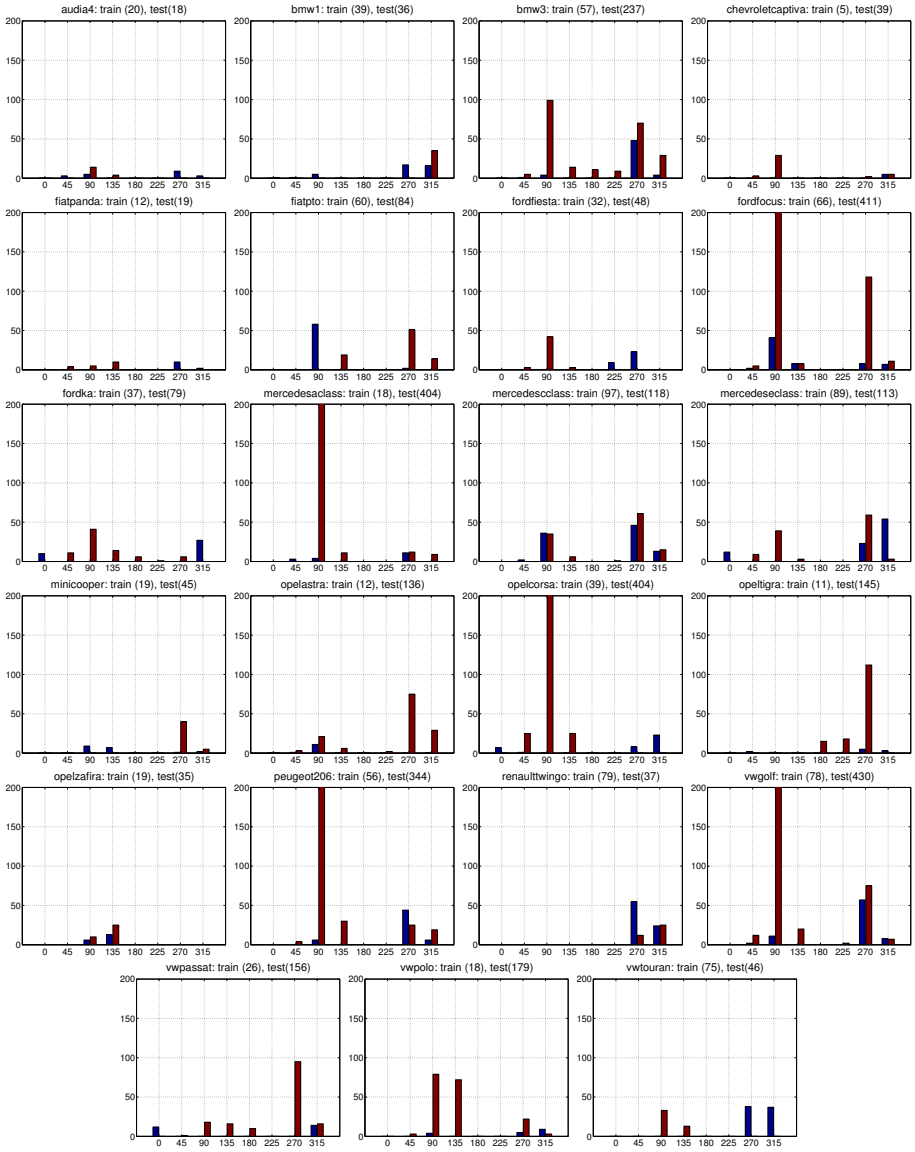


Figure 6: *Car-models* train and test statistics over 8 viewpoint bins.

test viewpoint data distributions tend to be skewed and sparse, especially on the fine-grained category levels, which, in addition to the low numbers of training examples, represent serious challenges when learning models for the fine-grained categories.

7 Prior visualization in 3D

In this section we visualize in 3D the **SVM-Σ**, **SVM-MV** and the **SVM-Σ-NB2ALL** priors learned for the *car* class on the 3D object classes dataset [39]. For that purpose, we sample a cell (c, r) from viewpoint v from the target model, which we call *reference* cell, and back-project on a 3D CAD model the learned weights (dependencies) for that particular cell in the Σ_s (see Eq. (3) in the paper) correlation matrix ($K_s = \mathbf{I} - \lambda\Sigma_s$).

Figure 7 visualizes the dependencies for an example *reference* cell. On the top of the figure, an example target model of the *car* class, trained on the 3D object classes dataset [39] is shown, along with the cell for which the weights in the prior are visualized (denoted as red cell). As each cell has $L = 32$ dimensions, we average the dependencies across all of them. Each cell is back-

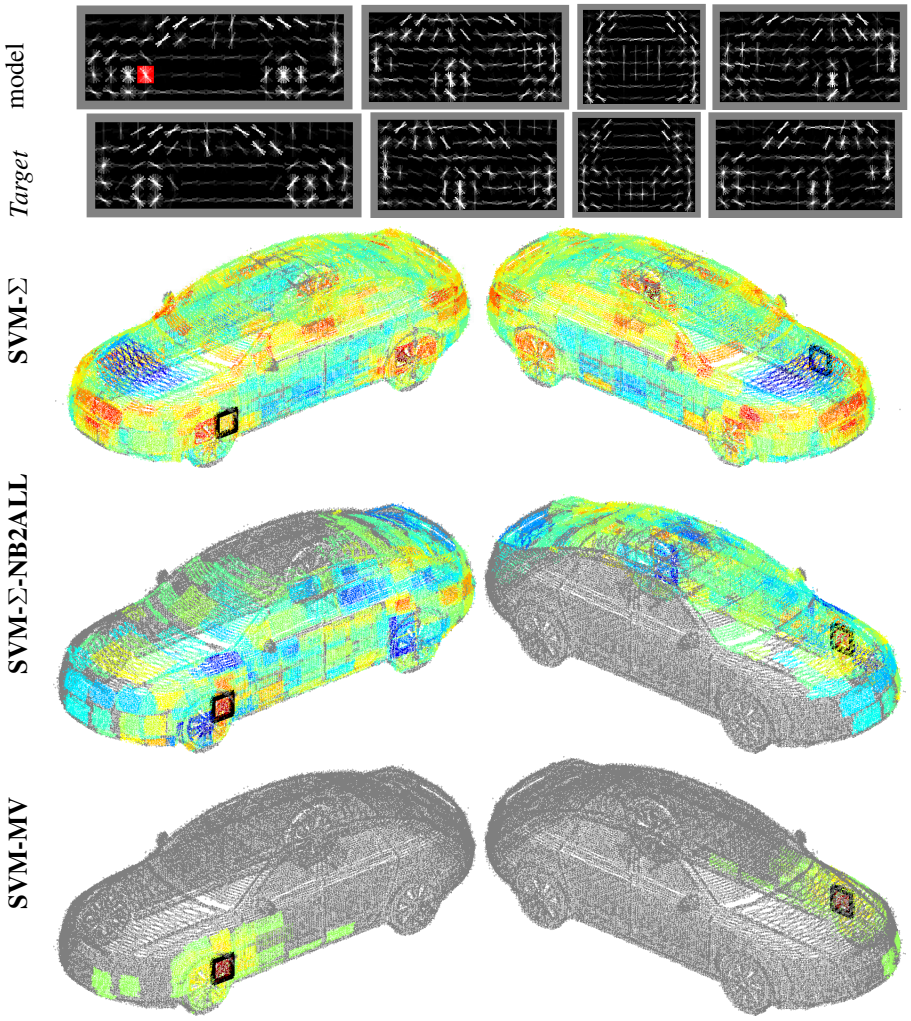


Figure 7: Prior visualization in 3D and target model (row 1). Red indicates the *reference* cell. Prior visualizations in 3D for the red cell: **SVM-Σ** (row 2), **SVM-Σ-NB2ALL** (row 3) and **SVM-MV** (row 4). The black cube indicates the *reference* cell back-projected into 3D.

projected to 3D by aligning the model template with the 3D CAD model. The alignment is two stage procedure involving viewpoint alignment in the first stage and template to rendered object alignment in the second stage. In Figure 7 rows 2, 3 and 4 visualize the cell dependencies for the **SVM-Σ**, **SVM-Σ-NB2ALL** and the **SVM-MV** priors, respectively (see Section 3 in the paper for details on the different priors). The dependencies on the 3D CAD model are visualized from two different views (left and right column). Red colors signify positive correlations, blue colors signify negative correlations. If the cells are not correlated (all dependencies are 0), gray color is used.

Observations. The **SVM-Σ** model (row 2) reveals symmetric structures in the object itself, which can be seen from the positive dependencies (red color) on the 4 wheels of the *car* for the *reference* cell. **SVM-Σ-NB2ALL** (row 3) and **SVM-MV** (row 4) establish dependencies with only a few cells in the model (lots of gray points) as they are both restricted to learning dependencies among neighboring views (**SVM-Σ-NB2ALL**) or neighboring cells (**SVM-MV**) only.