Detection and Identification of Rare Audiovisual Cues

Insperata accident magis saepe quam quae speres.
(Things you do not expect happen more often than things you do expect) Plautus (ca 200(B.C.)

Project no: 027787

DIRAC

Detection and Identification of Rare Audio-visual Cues

Integrated Project
IST - Priority 2

DELIVERABLE NO: D4.7
Automatic object detection from small sample and optimization, the perception of inconsistent events

Date of deliverable: 31/12/07
Actual submission date: 1/2/08

Start date of project: 01.01.2006 Duration: 60 months

Organization name of lead contractor for this deliverable: **HUJI**

Project co-funded by the European Commission within the Sixth Framework Program (2002-2006)		
Dissemination Level		
PU	Public	X
PP	Restricted to other program participants (including the Commission Services)	
RE	Restricted to a group specified by the consortium (including the Commission Services)	
CO	Confidential, only for members of the consortium (including the Commission Services)	

DIRAC
Detection and Identification of Rare Audiovisual Cues

Insperata accident magis saepe quam quae speres.
(Things you do not expect happen more often than
things you do expect) Plautus (ca 200(B.C.)

D4.7 AUTOMATIC OBJECT DETECTION FROM SMALL SAMPLE AND OPTIMIZATION

The Hebrew University of Jerusalem (HUJI)

Abstract:

We investigated the computational properties of natural object hierarchy in the context of constellation object class models, and its utility for object class recognition. We first observed an interesting computational property of the object hierarchy: comparing the recognition rate when using models of objects at different levels, the higher more inclusive levels (e.g., Closed-Frame Vehicles or Vehicles) exhibit higher recall but lower precision when compared with the class specific level (e.g., bus). These inherent differences suggest that combining object classifiers from different hierarchical levels into a single classifier may improve classification, as it appears like these models capture different aspects of the object. We describe a method to combine these classifiers, and analyze the conditions under which improvement can be guaranteed. When given a small sample of a new object class, we describe a method to transfer knowledge across the tree hierarchy, between related objects. Finally, we describe extensive experiments using object hierarchies obtained from publicly available datasets, and show that the combined classifiers significantly improve recognition results.

Exploiting Object Hierarchy: Combining Models from Different Category Levels

Abstract

We investigated the computational properties of natural object hierarchy in the context of constellation object class models, and its utility for object class recognition. We first observed an interesting computational property of the object hierarchy: comparing the recognition rate when using models of objects at different levels, the higher more inclusive levels (e.g., Closed-Frame Vehicles or Vehicles) exhibit higher recall but lower precision when compared with the class specific level (e.g., bus). These inherent differences suggest that combining object classifiers from different hierarchical levels into a single classifier may improve classification, as it appears like these models capture different aspects of the object. We describe a method to combine these classifiers, and analyze the conditions under which improvement can be guaranteed. When given a small sample of a new object class, we describe a method to transfer knowledge across the tree hierarchy, between related objects. Finally, we describe extensive experiments using object hierarchies obtained from publicly available datasets, and show that the combined classifiers significantly improve recognition results.

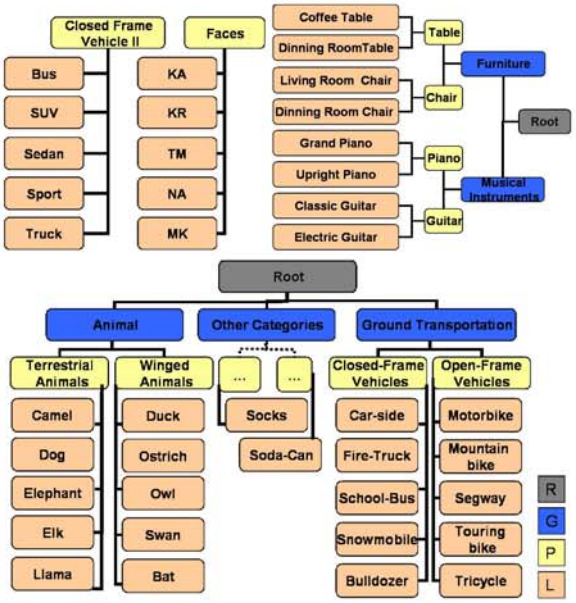

Figure 1. The four hierarchies used in our experiments. Categories at different levels of the tree are labeled (and color-coded) as follows: 'L' denotes the Leaf level, 'P' denotes the Parent level, 'G' denotes the Grandparent level, and 'R' the level of all objects. Our largest hierarchy (lowest diagram above) contains object classes from the CalTech256 database [8]. The facial hierarchy contains objects from [12].

1. Introduction

Human cognition relies on a hierarchal representation of objects in the world (see examples in Fig. 1), in the process of recognizing and referring to objects. How can we use such hierarchical structure to improve object recognition and categorization? This question has been addressed in a number of recent papers, mostly pursuing different directions to exploit this hierarchy when confronted with new categories (the small sample problem). The directions under study included the transfer of knowledge from known to new categories using Bayesian priors [11], sharing parts between objects at different levels of the hierarchy and improving generalization [13, 3], learning distance functions using related classes [6, 9], and transferring features [10, 4] or structure [5] from known classes to new ones. Often, when working on object class recognition, objects are represented by parts (or features) learned directly from example images of each object category, where relations between the parts (geometrical and possibly other) may sometimes be captured by graphical models trained using the same data.

We address the question posed above from a somewhat different point of view. We observe that the natural object hierarchy offers at our disposal a rich family of classifiers, for each category in each node of the hierarchy tree; the similarity between these classifiers varies, possibly in some relation to the distance between them on the object tree, but they all share some common characteristics. For example, if we build these classifiers with a discriminative algorithm that uses the background images of the CalTech256

database [8], then all the classifiers are trained to distinguish a certain isolated object from a background of clutter. Such commonalities may permit the combination of different classifiers to improve the recognition of specific object classes. We expect this improvement to be more pronounced when using related objects, and in particular objects from higher (inclusive) levels in the hierarchy tree.

The idea of combining classifiers has been extensively studied in the machine learning community under different frameworks, including committee machines, ensemble average, or boosting. In light of the so-called bias/variance dilemma [7], the ensemble average of a number of classifiers should improve generalization, as long as the classifiers are both accurate and diverse. One common way to obtain such a collection of classifiers is to train different classifiers with different samples of the training data. But note that the object recognition classifiers, trained to recognize different objects at different levels of the object hierarchy tree, may be viewed as just such - classifiers trained on different resamples of the training data. Viewed this way, we may expect to see improvement when combining any set of classifiers, even those trained to recognize very distinct objects. At the same time, for obvious reasons, we expect to see larger improvement when combining classifiers trained to recognize similar objects (those closer to each other on the tree), as compared to very different ones.

Thus the main conceptual contribution of this paper is to identify the object hierarchy as a source of classifier variability, which is induced by different and inherently meaningful resampling of the training data. We then go ahead and describe a framework to combine the classifiers linearly, using each classifier's probabilistic output and the corresponding LRT (Loglikelihood Ratio Test) value. This approach is somewhat different, and possibly more powerful, than the traditional ensemble of classifiers, since each object classifier builds a different representation of the data based on the training subset that it sees. The approach differs from boosting and bagging in that the data resampling is not based on some bootstrapping procedure, but on supervised information given to the system via the object hierarchy tree. Clearly our approach could be augmented with boosting to further improve results.

The rest of this paper is organized as follows. First, we review in Section 2 the object class constellation model used to obtain each object classifier, and describe how we combine these classifiers. We also discuss the theory underlying our approach. In Section 4 we show that a number of intuitive object hierarchies (described in Section 3), provided by a human teacher, reveal consistent and sensible computational characteristics. Specifically, classifiers built for more specific objects (such as 'bus') - corresponding to the lowest level in the object tree, are characterized by high precision (or high specificity) and low recall (low sensitivity), while classifiers built for more general classes of objects at the next levels up the tree (such as 'vehicle') show the opposite - low specificity and high sensitivity. In general, the specificity of object classifiers decreases as we ascend the object tree. Then, In Section 5, we describe how to use the hierarchy to transfer knowledge between classes, as a way to address the small sample problem. Finally, we investigate in Section 6 combined classifiers as a general framework for the construction of object recognizers. We show empirically that combined classifiers improve performance significantly (over all constituent classifiers), and that typically three-level combinations perform better still than two-level combinations.

2. Combining Object Models

Our object class model To learn object models, we use the method described in [2], because its computational efficiency allows us to consider models (and images) with many features. The algorithm learns a generative relational part-based object model, modeling appearance, location and scale. Location and scale are relative to the unknown object location and scale, as captured by a star-like Bayesian network. The model's parameters are discriminatively optimized using an extended boosting process. This model has been shown to achieve competitive recognition results on standard benchmark datasets, approaching the state-of-the-art in object class recognition. Thus we believe that the results and improvements we show are general, and can be replicated with other, conceptually similar, part-based models.

Based on this model and some simplifying assumptions, the likelihood ratio test function is approximated (using the MAP interpretation of the model) by

$$F(\mathbf{x}) = \max_{C} \sum_{k=1}^{P} \max_{u \in Q(\mathbf{x})} \log p(u|C, \theta^k) - \nu \qquad (1)$$

with P parts, threshold ν, C denoting the object's location and scale, and $Q(\mathbf{x})$ the set of extracted image features.

Classifier combination rule In our experiments we combined 2, 3 and 4 object classifiers. For each classifier, we used its LRT value from (1), obtaining a 2-, 3- or 4-dimensional vector respectively. We then trained a Support Vector Machine classifier with the same training data represented in this new vector space, using a linear kernel.

The bias/variance dilemma Let $\mathbf{x} \in \mathcal{X}$ denote the image, $F(\mathbf{x})$ the LRT output of the classifier from (1), and $D(\mathbf{x})$ denote the binary random variable assigning 1 to class images, and -1 to background images. It can be readily

shown that the mean-square error between classifier F and the desired output D can be decomposed as follows:

$$\begin{aligned} E[(F(\mathbf{x}) - E[D(\mathbf{x})])^2] &= B(F(\mathbf{x})) + V(F(\mathbf{x})) \\ B(F(\mathbf{x})) &= (E[F(\mathbf{x})] - E[D(\mathbf{x})])^2 \\ V(F(\mathbf{x})) &= E[(F(\mathbf{x}) - E[F(\mathbf{x})])^2] \end{aligned}$$

where $B(F(\mathbf{x}))$ denotes the classifier's bias, and $V(F(\mathbf{x}))$ its variance. It can also be shown that when considering an ensemble average of such classifiers, the bias of the new classifier remains the same as the bias of $F(\mathbf{x})$, but its variance is reduced. As a result, the mean square error of the new classifier is also reduced.

The sensitivity/specificity tradeoff We now analyze the case of two classifier combination, where one classifier has high sensitivity and low specificity and the other has low sensitivity and high specificity. Recall that this is the computational property that distinguishes object classifiers from lower levels of the object hierarchy tree and classifiers from higher levels of the tree (see Section 4), and thus this analysis is revealing.

Given the function $F(\mathbf{x})$ from (1), define the classifier $F^*(\mathbf{x}) = sign(F(\mathbf{x}))$[1]. Let $F_1(\mathbf{x}), F_2(\mathbf{x})$ denote two classification functions, and $G(\mathbf{x}) = \frac{F_1(\mathbf{x})+F_2(\mathbf{x})}{2}$ its ensemble average. Let $G^*(\mathbf{x}) = sign(G(\mathbf{x}))$ denote the corresponding classifier, and let $G^{**}(\mathbf{x}) = \frac{F_1^*(\mathbf{x})+F_2^*(\mathbf{x})}{2}$ denote another related classifier.

We compute the error probability P_E of classifier $G^*(\mathbf{x})$:

$$\begin{aligned} &4P_E(G^*(\mathbf{x})) = E[(G^*(\mathbf{x}) - D(\mathbf{x}))^2] \qquad (2) \\ &= E[((G^*(\mathbf{x}) - G^{**}(\mathbf{x})) + (G^{**}(\mathbf{x}) - D(\mathbf{x})))^2] \\ &= E[(G^* - G^{**})^2] + E[(\tfrac{1}{2}(F_1^* - D))^2] \\ &+E[(\tfrac{1}{2}(F_2^* - D))^2] + 2E[(\tfrac{1}{2}(F_1^* - D))(\tfrac{1}{2}(F_2^* - D))] \\ &+2E[(G^* - G^{**})]\{E[\tfrac{1}{2}(F_1^* - D)] + E[\tfrac{1}{2}(F_2^* - D)]\} \end{aligned}$$

Note that

$$E[(\tfrac{1}{2}(F_i^*(\mathbf{x}) - D(\mathbf{x})))^2] = P_E(F_i^*)$$

and

$$\begin{aligned} E[\tfrac{1}{2}(F_i^*(\mathbf{x}) - D(\mathbf{x}))] &= P[F_i^*(\mathbf{x}) = 1, D(\mathbf{x}) = -1] \\ &-P[F_i^*(\mathbf{x}) = -1, D(\mathbf{x}) = 1] = \Delta S(F_i^*) \end{aligned}$$

where $\Delta S(F_i^*)$ denotes the classifier's preference to either recall (sensitivity) or precision (a measure typically similar to specificity)[2], i.e., its sensitivity minus its specificity.

[1]For convenience, we define the sign function as $sign(F) = 1$ if $F \geq 0$, and $sign(F) = -1$ if $F < 0$.

[2]Notation reminder: *recall* and *sensitivity* denote the rate of true positives, *specificity* denotes the rate of true negatives, and *precision* denotes the fraction of true positives among all examples identified as positive.

Henceforth we shall call $\Delta S(F_i^*)$ the 'recall/precision primacy'. Finally,

$$\begin{aligned} E[(G^* - G^{**})^2] &= P(F_1^*=1,F_2^*=-1) + P(F_1^*=-1,F_2^*=1) \\ &\leq P_E(F_1^*) + P_E(F_2^*) \end{aligned}$$

(with equality only when F_1^*, F_2^* err on disjoint sets of examples).

Putting all the above together, we get

$$\begin{aligned} 4P_E(G^*) \leq & 2P_E(F_1^*) + 2P_E(F_2^*) + 2\Delta S(F_1^*)\Delta S(F_2^*)] \\ & +2E[(G^* - G^{**})](\Delta S(F_1^*) + \Delta S(F_2^*)) \\ & -2\rho(\tfrac{1}{2}(F_1^* - D), \tfrac{1}{2}(F_2^* - D)) \qquad (3) \end{aligned}$$

where $\rho(X,Y) = E[XY] - E[X]E[Y]$ denotes the non-normalized correlation coefficient of X, Y.

We can now state our main result:

Result: Assume that F_1^*, F_2^* are two classifiers with opposite recall/precision primacy, i.e. and w.l.o.g., $\Delta S(F_1^*) \geq 0$ and $\Delta S(F_2^*) \leq 0$; thus $\Delta S(F_1^*) \cdot \Delta S(F_2^*) \leq 0$. Assume further that the magnitude of their primacy is similar, i.e., $|\Delta S(F_1^*)| \approx |\Delta S(F_2^*)|$, and that their correlation with respect to the data is small, i.e., $|\rho(\frac{1}{2}(F_1^* - D), (\frac{1}{2}(F_2^* - D)))| < |E[\frac{1}{2}(F_1^* - D)]E[\frac{1}{2}(F_2^* - D)]|$. Then it follows from (3) that

$$P_E(G^*) \leq \frac{P_E(F_1^*) + P_E(F_2^*)}{2}$$

In other words, the error probability of the combined classifier is smaller (usually significantly so) than the mean error probability of the constituent classifiers.

In practice, we see that the combined error is typically smaller than the minimal error of the constituent classifiers with opposite recall/precision primacy, see Section 6.

3. Datasets and General Experimental Setup

Datasets

In our experiments, we used an extensive data set containing various objects that can be found in natural scenes. As much as possible, classes were taken from standard benchmark datasets, with a few exceptions (to be detailed shortly). We organized these objects into four natural hierarchies. Examples from the object classes and background images can be viewed in Fig. 2. A summary of the hierarchies is provided in Fig. 1.

In the rest of this paper we use the following notation to refer to object classes at different levels in the hierarchy (see Fig. 1): specific object classes, like 'Elk' and 'Tricycle', are labeled 'L' (for Leaf). More inclusive categories, like 'Terrestrial Animals', are labeled 'P' (for Parent). Categories at

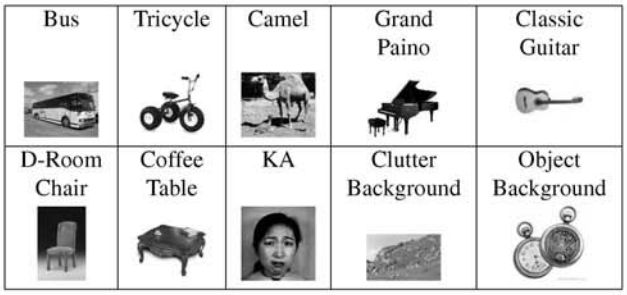

Figure 2. Examples taken from the object classes and background images, used to train and test our different Category level models, see Fig. 1.

the next level up, like 'Animals', are labeled 'G' (for Grandparent). Finally, the category of all objects is denoted 'R' (for Root).

For the discriminative learning procedure (see Section 2) and in order to evaluate the recognition results, we used two types of background. When using classes from the CalTech256 dataset [8] (the lowest hierarchy in Fig. 1), we used their Clutter Background images as well. With the remaining 3 smaller hierarchies and to achieve greater variability (and additional challenges) in the conditions of our experiments, we used our own Object Background dataset, containing various images of objects different from the learnt objects. This background was manually collected using Google, PicSearch and online catalogues.

CalTech256 Object Hierarchy This hierarchy includes pictures of various objects from the CalTech256 dataset [8], see Fig. 1. We chose objects that can be naturally organized into a sensible hierarchy: 'Animals' - including 'Terrestrial Animals' and 'Winged Animals', and 'Ground Transportation' - including 'Open-Frame Vehicles' and 'Closed-Frame Vehicles'. Models were learnt for objects from the 'Terrestrial Animals' and 'Open-Frame Vehicles' classes; other objects were used for training 'G' and 'R' level models. The CalTech256 Clutter Background was used with this dataset. We note that our category affiliations may not be identical to those used in [8], in an attempt to emphasize visual similarities over functional (thus ignoring, for example, the motorized vs. un-motorized distinction); we also used different names for the inclusive categories. The images in the Ground-Transportation Category were flipped to achieve uniform orientation (all vehicles pointing rightwards).

Closed-frame VehicleII Hierarchy This hierarchy contains 5 classes of common vehicles (more so than those in the CalTech256 database), contrasted with pictures from the 'Object Background'. Pictures (for the vehicle classes and background) were chosen manually from Google and PicSearch, showing vehicles at similar canonical orientation.

Faces Hierarchy This hierarchy contains pictures of 5 individuals taken from [12], with varying facial expressions.

Basic-Level Hierarchy This hierarchy was built to match standard hierarchies favored in the cognitive science literature, representing canonical categorization levels (basic-level, sub-ordinate, and super-ordinate). Pictures were obtained from the CalTech101 subset of [8], or collected using mainly online shopping catalogues.

General experimental setup

In general, we *always* tested recognition performance for specific object categories from level 'L' (leaf) of all the respective hierarchies. Thus, for example, when comparing three models such as 'Llama', 'Animal', and 'Terrestrial Animal', they were all tested on the recognition of Llama pictures.

For each hierarchy, all models were trained with the same background images but different object images. All tests were done with the same background and test images, and the same algorithm parameters. Each experiment was repeated 60-100 times, with new random samples of train and test images. Since the Equal Error Rate (denoted EER) of the ROC is not well suited when the number of positive examples is much smaller than the number of negative examples [1], we used the preferred EER of the Recall Precision Curve (RPC).

To compare performance, we report two measures: (i) Precision and Recall of each classifier; (ii) EER of the RPC curve for each classifier, computed by varying the threshold of the optimal linear SVM classifier.

4. Object Hierarchy

We study the computational properties distinguishing objects from different levels in the object hierarchy tree, revealing opposite recall/precision primacy - high precision for the lowest level (specific) models, and high recall for higher level (inclusive) models. With sufficiently large samples per object, and given that we always test the recognition of object classes from level 'L', not surprisingly object models from level 'L' show superior recognition performance. In accordance, we see a decrease in performance as we use models ascending the object hierarchy tree.

Experimental setup We tested the recognition of each specific object from level 'L' by 5 types of models learnt using object categories from different levels in the hierarchy tree, see Table 1. To assure fair comparison, all models saw the same train images of the 'L' object they were tested on; an illustrative example of this procedure is shown in Table 1. In different experiments we varied the number of train images per 'L' object: 5, 10, 15, 20, 25 and 30. Three

hierarchies were used: CalTech256 Object, Closed-frame VehicleII, and Faces.

Exp	Category training set L P G R DB	Example: Llama
1	1	Llama
2	5	Llama, Camel, Dog, Elk Elephant
3	5	Llama, Duck, Owl Swan, Ostrich
4	5	Llama, Soda-can, Sock Segway, Motorbike
5	5	Llama, Segway, Tricycle Motorbike, Mountainbike

Table 1. This table shows the 5 different models learnt and evaluated on the recognition of 'L' level objects. 'DB' refers to a 'G'-level category from a Different Branch of the tree. Examples are shown for the Llama as 'L' level object. In each different experiment, each 'L' class provided the same fixed amount of pictures to the training set (5, 10, 15, 20, 25 or 30.)

With all the 3 hierarchies, we used test data composed of images of the target object and images of the relevant background in equal proportion. None of the test images was used for training. With the CalTech256 Object Hierarchy, where the models are learnt using the Clutter Background, we also conducted additional experiments, using for test data images from the target 'L' object mixed with images of a different 'L' object.

4.1. Results

Recall and Precision are shown in Fig. 3, comparing recognition when using 'L' and 'P' level models (left), and 'L' and 'G' level models (right). In the first comparison (Leaf vs. Parent), only 4 representative examples from the 3 hierarchies are shown; very similar results were obtained with all other objects. In the second comparison (Leaf vs. Grandparent), all objects from the CalTech256 Object Hierarchy are shown. These graphs clearly show the Recall/Precision Primacy effect, where 'L' models show high precision and low recall in recognition, while 'P' and 'G' models show high recall and low precision. This happens for all objects, regardless of the number of training examples.

The Recall Precision Curve and its corresponding EER are shown in Fig. 4 for two representative examples. Not surprisingly, we see that with sufficient training examples per 'L' object (as for the Dog class), the 'L' model performs best, and performance deteriorates as we ascend the object hierarchy. As the sample per class decreases, the advantage of the 'L' model over the 'P' model decreases, eventually the 'P' model might outperform the 'L' model. Once again, similar phenomenon is observed for all objects.

Looking more closely at these results, we see that the

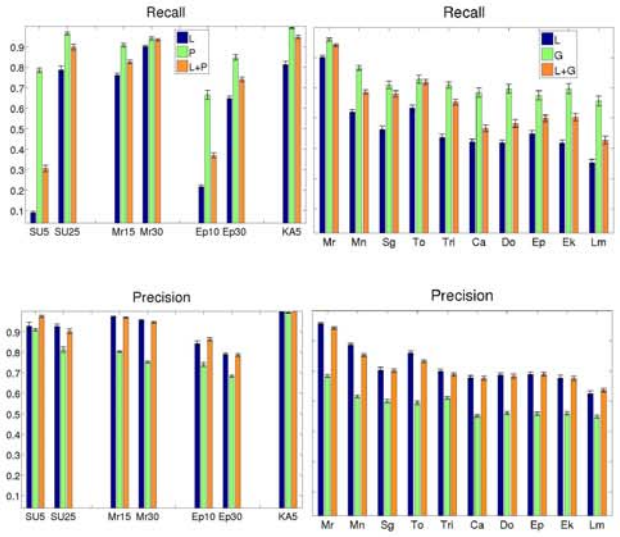

Figure 3. Recall and Precision of classifiers. Left column: four representative object classes, recognized with the 'L', 'P' and the combined 'L+P' models. 'SU' refers to the SUV class, 'Mr' - Motorbike, 'Ep' - Elephant, 'KA'- one of the female faces. The numbers indicate the size of the train set (e.g., 'SU5' refers to the SUV model trained with 5 SUV examples). Right column: all learnt object classes from the CalTech256 Object Hierarchy trained with 30 images per object class, and recognized with the 'L', 'G' and the combined 'L+G' models.

recall/precision primacy difference occurs regardless of the overall recognition rate. Specifically, for the Elephant with 10 training examples, we see from Fig. 4 that the 'P' model performs better than the 'L' model, and vice versa for the Elephant with 30 training examples. Still, Fig. 3 shows the same Recall/Precision primacy in both cases.

5. Using hierarchy to transfer knowledge

We study here how to transfer information between related objects, located nearby in the object hierarchy tree, to handle the problem of small sample or the appearance of new objects.

Experimental setup We tested the recognition of each specific object from level 'L' by 7 types of models learnt using object categories from different (more inclusive) levels in the hierarchy tree, see Table 2, which also shows an illustrative example of this procedure. The test background set consisted of 75 background images, while the test object set consisted of 30 images. Only the Basic-Level Hierarchy was used.

5.1. Results

Fig. 5 shows the results. Clearly the 'P' class model transfers information most effectively (seen in the superi-

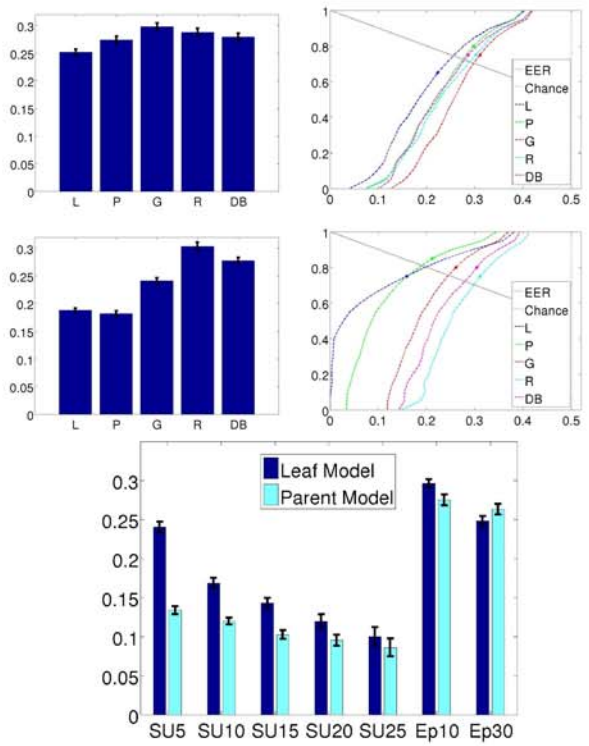

Figure 4. Performance of the different category level models. Top & middle: left column - the EER of the RPC, right column - full RPC curve where the point of the original classifier is highlighted . Once again, 'L' denotes the Leaf category, 'P' - Parent, 'G' - GrandParent, 'R' - Root, and 'DB' - Different Branch. Top: performance of all models tested on the 'Dog' class from the Caltech256 hierarchy. Middle: same for the 'Mountain-Bike' class from the Caltech256 hierarchy. Bottom: performance of the 'L' and 'P models on the 'SUV' class from the 'Closed-Frame VehiclesII' hierarchy and on the Elephant class from the 'Terrestrial Animals'. EER scores are shown as a function of the size of the training set - increasing from 5 up to 25 for the SUV, sizes 10 and 30 for the Elephant. Note the decrease in the performance superiority of the SUV 'P' model over the 'L' model till it is insignificant, as the train size increases. Note the opposite superiority of 'L' vs. 'P' models when comparing the two Elephant class models.

ority of Exp. 3 over Exp. 5-7), and improves performance over the small sample case (seen in the superiority of Exp. 4 over Exp. 2).

5.2. Discussion

The results above show a clear hierarchy structure, where models which are learnt from nearby objects (brothers) in the object hierarchy tree can substantially improve the recognition results of each other. It shows the possibility for the success of a learning-to-learn scheme - where fewer examples of the goal object class are used in the learning process, augmented by examples from different related classes.

Ex	Object training set					Example:
	L	P	G	DB	BG	Classic Guitar
1	35					Classic Guitar
2	1					Classic Guitar
3		30				Electric Guitar
4	1	30				Classic, Electric Guitar
5			30			Grand Piano
6				30		Living Room Chair
7					30	Background

Table 2. The models learnt in the different experiments on the transfer of information between classes. 'L' refers to the Leaf level, 'P' to the Parent, 'G' to the Grandparent, 'DB' to a Different Branch of the tree, and 'BG' to the background.

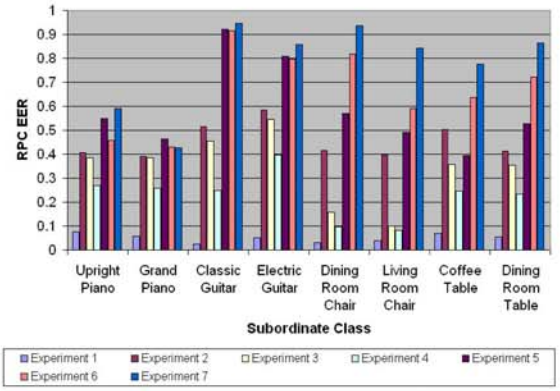

Figure 5. Transfer of knowledge between related classes. Description of the different experiments is given in Table 2.

6. Using hierarchy to improve classification via combination

We now ask whether the combination of two or more object model classifiers, which are based on different category levels, can improve the performance of the original classifiers.

Experimental setup We used the same setup, same data, and same learnt models as used in Section 4. We tested different combinations by combining two or more models from different category levels as described in Section 2. The different combinations that we studied are summarized in Table 3.

For comparison, we used a naïve alternative method, which learned directly an object model using the same set of images as used by the combined model. Each image in this set was initially weighted to reflect its true weight on the combined model. For example, when combining two models such as 'Llama' and 'Terrestrial Animals', we note that the Llama images provided all the training set for the Llama model, and only 20% of the training set for the 'Ter-

Exp	Example: Llama as Leaf Level
L+P	'Llama' + 'Terrestrial Animals'
L+G	'Llama' + 'Animals'
L+R	'Llama' + 'Tree Root'
L+DB	'Llama' + 'Open-Frame Vehicles'
P+G	'Terrestrial Animals' + 'Animals'
L+P+G	'Llama' + 'Terrestrial Animals' + 'Animals'
L+P+G+R	'Llama' + 'Terrestrial Animals' + 'Animals' + 'Tree Root'

Table 3. The different combinations we studied: '+' denotes a combination of two models. 'L' refers to the Leaf level, 'P' - Parent, 'G' - Grandparent, 'R' - Root, and 'DB' - Different Branch.

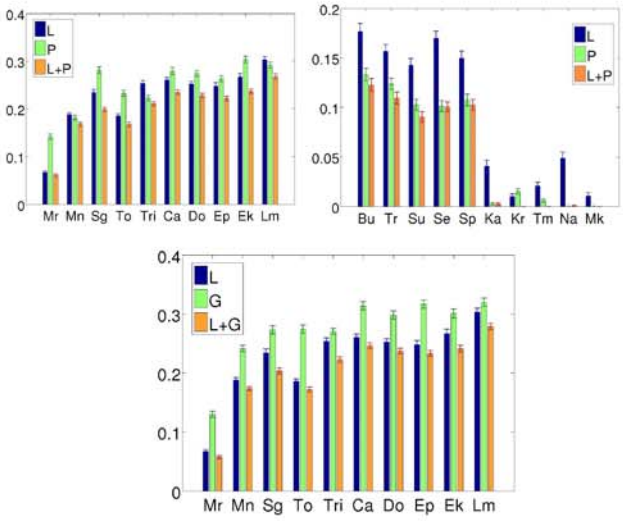

Figure 6. Comparison of combined and original classifiers. 'L' denotes a leaf model, 'P' - parent model, 'G' - grandparent model, 'L+P' leaf/parent combined model and 'L+G' leaf/grandparent combined model. Top: object models recognized with the 'L', 'P' and the combined 'L+P' models. Bottom: object models recognized with the 'L', 'G' and the combined 'L+G' models. Top-Left: CalTech256 Object Hierarchy, 'Open-frame Vehicle' and 'Terrestrial Animal', with 30 training images per object class. Specifically, 'Mr' denotes the Motorbike class, 'Mn' - Mountain-Bike , 'Sg' - Segway, 'To' - Touring-Bike, 'Tri' - Tricycle, 'Ca' - Camel, 'Do' - Dog, 'Ep' - Elephant, 'Ek' - Elk, 'Lm' - Llama. Top-Right: 'Closed-Frame VehicleII' Hierarchy with 15 training images per object class, and 'Faces' Hierarchy with 5 training images per object class. Bottom: CalTech256 Object Hierarchies, with 30 training images per object class

restrial Animals' model; thus in the training of the combined model, the Llama training set received total weight of 0.6, while the remaining 4 classes received total weight of 0.4. The background train set remained unchanged.

6.1. Results

Fig. 6 shows the EER recognition results for all 20 objects, from the 'P' classes of 'Terrestrial Animals', 'Open-

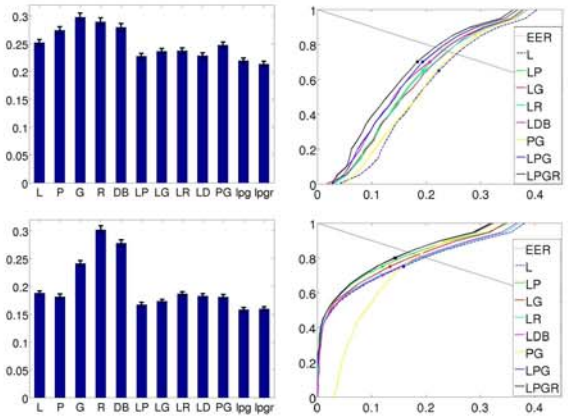

Figure 7. All different combinations of classifiers. Left: the EER of the RPC. Right: full RPC (Recall Precision Curves). Top: results when recognizing the 'Dogs' class. Bottom: results with the 'Mountain Bike' class.

Frame Vehicles', 'Closed-Frame VehiclesII', and 'Faces'. We show recognition results with 3 models - 'L', 'P', and the combined 'L+P', fixing the number of training examples to 30, 30, 15, and 5 for each object in the 4 'P' classes respectively. We also show recognition results with 3 models - 'L', 'G', and the combined 'L+P', with 30 training examples and two classes of the CalTech256 Object Hierarchy.

Clearly, almost always, the combined model performed better than both constituent models. This happened for all objects and all training conditions, regardless of which of the constituent models was initially superior. The only exception occurred in the experiments with only 5 training images per object class (small sample). Moreover, in all experiments the combined model improved significantly the weak measure (either Recall or Precision) of each of the constituent models, as demonstrated in Fig. 3.

Fig. 7 shows results with the 7 different classifier combinations, listed in Table 3, for two object classes. These are representative results - similar results were obtained with all other classes. Note that the two-level combinations that obtain the highest performance are either the 'L+P' or 'L+G'. Not surprisingly, therefore, the best results are obtained with the three- and four-level combinations ('L+P+G' and 'L+P+G+R' respectively).

Fig. 8 shows the EER of the RPC in the second test condition, when test examples included an equal number of images from the target object and another unrelated distractor object (instead of the standard background images). Not surprisingly, when the 'L' model was combined with a model whose training set included pictures of the distractor object, the performance of the combined model was reduced. However, interestingly enough, this reduction is rather slight (see Fig. 8). This decrease remains slight even when the 'L' model is combined with a very poor classifier (under these conditions), like the 'R' or 'DB' ones. Thus

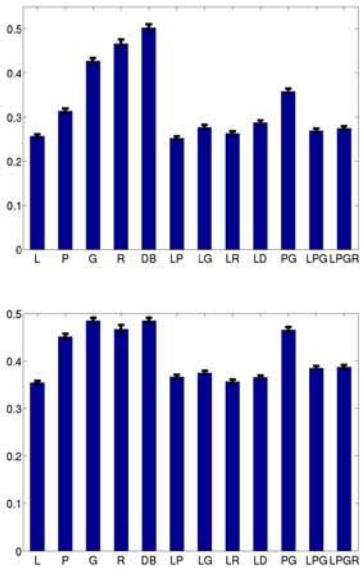

Figure 8. The EER of the RPC when the test examples included images of the target object - the 'Mountain Bikes' - and another object (instead of the standard background images). Top: models tested against 'Camel'. Bottom: models tested against 'Tricycle'.

these results seem to suggest that the improvement obtained by the combined classifier against the standard background is not accomplished at the high cost of reducing the discriminability of the new classifier against other, possibly related, objects.

Finally, we note that in most cases, the comparison against the naïve model showed superior results for the combined model, while in the other cases the advantage of the naïve model was not significant. On the other hand, the training time of the naïve model was substantially longer. Moreover, this training procedure is not modular, while the combination scheme we described is rather flexible.

7. Summary and Discussion

We analyzed the computational properties of constellation object class models, built to describe object categories at different levels of the object hierarchy tree. An interesting observation emerged, when comparing specific object models, trained using images of objects corresponding to the leaves of the hierarchy tree, with models built to describe categories at higher levels of the object hierarchy tree. The first (specific) models exhibit higher precision, while the second (inclusive) models exhibit higher recall. We provided the theoretical analysis showing why this situation should be favorable for the success of a classifier combined from two such constituents (one with higher precision, the other with higher recall), and demonstrated experimentally that significant improvement is indeed achieved in all cases. In our experiments the combined model performed better than all constituents models in almost all cases. The improvement magnitude was larger when the constituent classifiers corresponded to nearby objects in the hierarchy tree, showing that this improvement is not due simply to the larger training set.

In all our experiments, we used a specific part-based model that can be learned rather efficiently, and can therefore handle a relatively large number of parts (or features). Although we did not perform experiments to this effect, we believe that this improvement can be obtained with any object class model, and that the phenomena we have observed do not depend on the specific model we used.

References

[1] S. Agarwal and D. Roth. Learning a sparse representation for object detection. In *Proc. ECCV*, 2002. 4

[2] A. Bar-Hillel, T. Hertz, and D. Weinshall. Efficient learning of relational object class models. *Proc. ICCV*, 2005. 2

[3] A. Bar-Hillel and D. Weinshall. Subordinate class recognition using relational object models. *Proc. NIPS*, 19, 2006. 1

[4] E. Bart and S. Ullman. Cross-generalization: learning novel classes from a single example by feature replacement. *Proc. CVPR*, pages 672–679, 2005. 1

[5] A. Ferencz, E. Learned-Miller, and J. Malik. Building a classification cascade for visual identification from one example. *Proc. ICCV*, pages 286–293, 2005. 1

[6] M. Fink. Object classification from a single example utilizing class relevance metrics. *Proc. NIPS*, 17, 2004. 1

[7] S. Geman, E. Bienenstock, and R. Doursat. Neural networks and the bias/variance dilemma. *Neural Comput.*, 4(1):1–58, 1992. 2

[8] G. Griffin, A. Holub, and P. Perona. Caltech-256 object category dataset. Technical Report UCB/CSD-04-1366, California Institute of Technology, 2007. 1, 2, 4

[9] T. Hertz, A. Bar-Hillel, and D. Weinshall. Learning distance functions for image retrieval. *Proc. CVPR*, 2, 2004. 1

[10] K. Levi, M. Fink, and Y. Weiss. Learning From a Small Number of Training Examples by Exploiting Object Categories. *LCVPR04 workshop on Learning in Computer Vision*, 2004. 1

[11] F. Li, R. Fergus, and P. Perona. One-shot learning of object categories. *IEEE PAMI*, 28(4):594–611, 2006. 1

[12] M. Lyons, S. Akamatsu, M. Kamachi, and J. Gyoba. Coding facial expressions with gabor wavelets. *Proc. ICAFGR*, pages 200–205, 1998. 1, 4

[13] E. Sudderth, A. Torralba, W. Freeman, and A. Willsky. Learning hierarchical models of scenes, objects, and parts. *Proc. ICCV*, 2005. 1

Antithesis Modeling for Object Recognition

We now introduce a novel framework for object categorization developed at ETH which is based on antithesis modeling. On the theoretical side, this framework provides a sound argument for summing evidence, legitimizing the common, often unjustified practice in the context of Hough transform based approaches. On the algorithmic side, we propose a compact parametric representation which allows for efficient search of local maxima by means of fix-point iterations. We show that in this proposed framework, robust detection can be achieved by fast nearest-neighbor matching on the feature level, as opposed to the more expensive soft-matching required in comparable approaches. In the following sections, we describe the algorithm's main ideas. For details, as well as a quantitative evaluation, we then refer to [ETH-1], where the method's performance is also compared to a state-of-the-art detector.

1.1 Antithesis Modeling

Many current object detection systems are based either on a feature transformation (such as the Hough transform), or on a sliding-window classification scheme. The former methods fall into the category of bottom-up systems, which try to predict high-level object information from low-level features. The latter test all possible hypotheses and validate them by actively sampling low-level features, i.e. using a top-down strategy. The initial formulation of our antithesis framework presented below also falls into the class of sliding window classifiers. However, as a result of the concrete choice of our object model, we are able to rewrite it as a Hough transform-based system. In other words, our concrete modeling choice makes the duality of sliding window and Hough transform based classifiers explicit.

The main idea of our approach is diametrically opposite to most current object detection approaches from the literature (e.g. [2, 3, 6]). Instead of trying to model the likelihood that a given feature distribution stems from an instance of the target object category, we are interested in the probability of the contrary hypothesis, namely that the feature configuration occurs accidentally. The aim is then to quantify the probability that the features arise given the following antithesis: the observed feature configuration **f** is a random phenomenon, i.e. not due to an object. If this probability is small, then the antithesis has to be rejected, which leads to the conclusion that an object is present. Thus, the goal is to express the probability $P(\mathbf{f} \mid H_0, \lambda)$ of the observed features **f** for a given object position λ and assuming that the antithesis H_0 holds.

The big advantage of this formulation concerns the common assumption of independence between features. Such an assumption is often required for computational feasibility, but it is not justified when we try to express the joint likelihood of a feature configuration belonging to an object, especially if features are sampled so closely that they overlap in the image. Consequently, current local feature-based object detection approaches mainly differ in the varying degree of feature independence they assume [2, 3, 6] and in the measures they take to compensate for the problems caused by that oversimplifying assumption.

Under our antithesis H_0, in contrast, the features are random and as such independent of each other. Hence, the probability factorizes $P(\mathbf{f} \mid H_0, \lambda) = \prod_i P(f_i \mid H_0, \lambda)$, and it remains to quantify the "randomness" of a single feature f_i. On the one hand, a feature is random if it cannot be explained by the object model. On the other hand, if a feature is consistent with the model, there is still a slight chance that this happened accidentally, i.e. with probability $\varepsilon < 1$. The two cases can be fused into a single expression

$$P(f_i \mid H_0, \lambda) = \varepsilon^{1(f_i \in O_\lambda)}$$

where $\mathbf{1}(f_i \in O_\lambda)$ stands for the indicator function, which is 1 if the feature f_i is consistent with the object O_λ at location λ, and 0 otherwise. In order to avoid having to make a hard decision, we relax the indicator function and replace it by a continuous function $C(f_i \mid \lambda)$, which expresses the confidence that feature f_i is consistent with the object model.

The validity of the antithesis can now be evaluated for every possible position of an object, as in a sliding window approach. As nearby positions tend to be similar (which results in similar feature probability), we are seeking for locations where the antithesis is locally most unlikely. Putting everything together, the task of object detection reduces to minimizing the following function

$$\lambda^* = \arg\min_\lambda P(\mathbf{f} \mid H_0, \lambda) = \arg\min_\lambda \prod_i \varepsilon^{C(f_i|\lambda)}$$

$$\lambda^* = \arg\min_\lambda \sum_i C(f_i \mid \lambda) \log \varepsilon$$

$$\lambda^* = \arg\min \sum_i C(f_i \mid \lambda)$$

where the term $C(f_i \mid \lambda)$ encodes the object model information, and the reliability term ε does not affect the position of the minimum and drops out. As we can see, the information provided by features is combined additively. This is important for the robustness of the detection system and will be discussed in the following section.

The above derivation defines the framework only in a generic form. In order to get a concrete algorithm, the confidence function C has to be specified to encode an object model. For the detailed derivation how that is done in our proposed approach, we refer to [ETH-1]. As we show there, our chosen representation in terms of a parametric mixture model allows us to predict the object center in a Hough-transform like fashion, so that recognition can be performed very efficiently by means of fix-point iterations.

1.2 Summation of Evidence

As one of our main contributions, our proposed antithesis model leads to a sound explanation of the common practice of summing evidence. This summation, while yielding empirically good performance [6, 7], has often been used without a convincing justification. A rather pragmatic view on this question of summing information (provided by features) or taking their product is given by Kittler et al. [4]. They point out that the sum is a more robust way to combine classifiers than using their product. From a more practical point of view, the implicit shape model (ISM) [6] has received considerable attention. This Hough transform based algorithm is formulated in a probabilistic framework as a marginalization over features

$$p(\lambda \mid I) = \sum_i p(\lambda \mid f_i) P(f_i \mid I)$$

where I denotes the observed image. The first term describes the occurrence distribution where the object center is expected, relative to the observed feature, whereas the second term measures the matching confidence. Object detection is then achieved by finding maxima of $p(\lambda \mid I)$. As such, the algorithm is very similar to ours. However, this formulation implies that only one or the other feature is observed, while in practice they are all observed simultaneously. The question is: how can the summation of evidence, which leads to the empirically observed robust performance, be justified else? We believe that our antithesis model gives a nice and sound argument to this practice. Moreover, it suggests using a confidence measure instead of the probability density. Although these two quantities are strongly related, choosing one or the other has a strong impact on the results, as we will show in Section 1.4.

Williams et al. [9] approach the problem from a different direction based on a model introduced by Sudderth et al. [8]. There, the joint distribution of the object and the features is considered. In order to make that model tractable, the features are considered to be independent such that the joint probability factorizes, i.e. $P(f_{1...}\ f_n, \lambda) = P(\lambda) \prod_i P(f_i \mid \lambda)$. They obtain a justification for summing the evidence using a first-order Taylor approximation. However, the independence assumption made in this model is not very appropriate. In reality, given the presence of an object there are often long-distance interactions among features, which are not modeled by their framework. Although our model does not exploit these dependencies either, we do not reason about the presence of an object, but about the absence of an object, in which case the independence assumption can more readily be justified.

1.3 Soft-Matching Strategies

Our object representation builds on local features, which have become a matured image representation. The basic concept is to extract features f_i at given locations l_i, which are then vector quantized and represented by a representative codeword w_i. Due to noise or quantization effects, it may happen that the closest codeword is not necessarily the best association. The robustness of the system can therefore be increased by allowing some degree of soft-matching. Hence, one single feature is matched to several codewords instead of only one, and several occurrences are generated. Such a procedure has the positive side effect that it increases the number of samples available to estimate the distribution. Matching features to codewords happens at two places, namely during learning and recognition, where both can use soft or hard matching. The advantage of hard matching is that it is much faster to perform at recognition time [5], which has to be traded off for system robustness. The interesting question here is: does soft-matching yield a benefit?

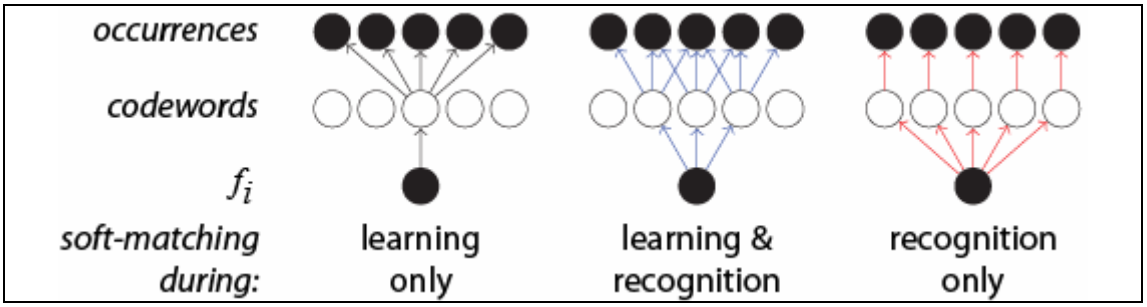

Figure 1. Soft-matching versus nearest neighbors. The black circles indicate feature values, whereas the white ones represent codewords. The single black circle at the bottom is a feature fi observed in a new test image. The features at the top are from the training set and are used to learn the occurrence distribution. One can see that the same activation radius can be obtained for all three variants, depending on the degree of soft-matching (1, 3, or 5-NN in this example) used during learning and recognition, respectively.

Interpreting feature matching in a probabilistic manner, the hard matching (i.e. NN-rule) corresponds to an activation distribution peaked on the best match. Soft-matching corresponds to a blurring of that (codeword) activation distribution. Consequently, we argue that applying soft-matching during learning and recognition corresponds to a double blurring, and a similar effect could be achieved by applying a stronger blurring either during learning or at recognition time (see Figure 1). As soft-matching yields more activations, it is preferably applied during learning. Since the resulting occurrence distribution will still be compressed by our parametric mixture model, this does not incur too much of an overhead. By shifting soft-matching to the learning stage, it becomes thus possible to use faster NN-matching during recognition, where speed is of prime importance. As we show in the attached paper [ETH-1], this strategy can be used without decreasing recognition performance.

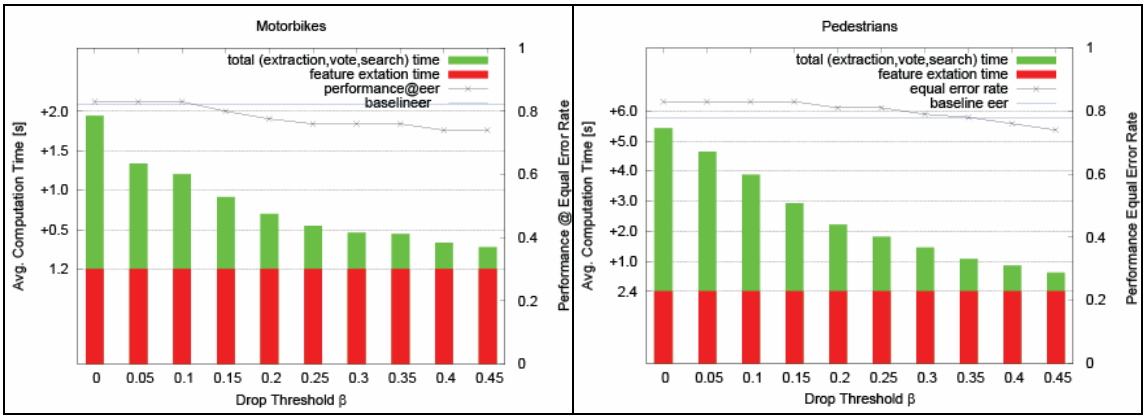

Figure 2. Detection time and performance at equal error rate with reduced occurrence distributions on two test sets. Feature extraction (in red) is a constant cost, which can be reduced dramatically with optimized detectors. With our proposed approach, the detection time can be significantly reduced at the cost of only a small decrease in accuracy.

1.4 Density versus Confidence

The probabilistic argument given in [6] defines the objective function to hypothesize object positions as a sum of probability densities. Our antithesis framework derived in Section 1.1, on the other hand, suggests the use of the confidence level associated with these densities. In [ETH-1], we have experimentally compared those two variants. Although the difference between the two objective functions seems actually fairly small, i.e. only a rescaling of each density function by a feature-dependent factor, an astonishing impact could be observed. Namely, the confidence based objective function suggested by the antithesis modeling clearly outperforms the density based variant.

1.5 Effect on Run-time

Finally, the parametric object model introduced in Section 1.1 offers an interesting opportunity for speeding up the proposed approach's run-time. Effectively, this model represents each feature's spatial distribution on the object as a mixture of Gaussians with a feature-dependent normalization factor. In order to speed up recognition, we can now drop all mixture components whose peaks are smaller than x% of the globally maximal peak. Figure 2 reports the resulting average detection time, as well as the achieved performance at equal error rate, for two test datasets of motorbikes and pedestrians. We see that for values up to x = 10% and 15%, respectively, the computation time (ignoring feature extraction) can be reduced by about half without decrease in accuracy. This yields a detection time of about 1.2s for a motorbike image (after feature extraction). If runtime is of prime importance, we can also sacrifice some accuracy in favor of speed. On the pedestrians, for example, we can increase the threshold to x = 35%, which yields the same equal error rate as the published baseline from [6]. Doing so, the detection time (after feature extraction) decreases from initially 5.5s to only 1.1s. Note that this simple, but very effective heuristic exploits the structure of our parametric model and would not be easily realizable in a non-parametric framework.

This result is especially interesting in conjunction with the fast feature detectors developed by KUL in WP1. With their help, the remaining feature extraction time of, on average, 1.2s and 2.4s for the motorbike and pedestrian images, respectively, can be dramatically reduced to only a few milliseconds. The proposed antithesis modeling framework thus promises a way to leverage that speed also for the later recognition stage.

References

[**ETH-1**] A. Lehmann, B. Leibe, L. Van Gool. "Antithesis Modeling for Object Recognition", submitted to *IEEE Conference on Computer Vision and Pattern Recognition (CVPR'08)*, Anchorage, USA, Oct. 2008. **(appended to this document)**

[2] D. Crandall, P. Felszenszwalb, D. Huttenlocher, "Spatial Priors for Part-Based Recognition using Statistical Models", in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR'05)*, 2005.

[3] R. Fergus, A. Zisserman, P. Perona, „Object Class Recognition by Unsupervised Scale-Invariant Learning", in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR'03)*, 2003.

[4] J. Kittler, M. Hatef, R. Duin, J. Matas. „On Combining Classifiers". In *IEEE Transactions on Pattern Analysis and Machine Learning*, Vol. 20(3), pp. 226–239, 1998.

[5] B. Leibe, K. Mikolajczyk, B. Schiele, „Efficient Clustering and Matching for Object Class Recognition", In *British Machine Vision Conference (BMVC'06)*, 2006.

[6] B. Leibe, A. Leonardis, B. Schiele, „Robust Object Detection with Interleaved Categorization and Segmentation", in *International Journal of Computer Vision*, Vol. 77, No. 1-3, 2008.

[7] A. Opelt, A. Pinz, A. Zisserman. „A Boundary-Fragment Model for Object Detection". In *European Conference on Computer Vision (ECCV'06)*, 2006.

[8] E. B. Sudderth, A. Torralba, W. T. Freeman, A. S. Willsky. "Learning Hierarchical Models of Scenes, Objects, and Parts". In *International Conference on Computer Vision (ICCV'05)*, 2005.

[9] C. K. I. Williams, M. Allan. "On a Connection Between Object Localization with a Generative Template of Features and Pose-Space Prediction Methods". *Technical Report* 0719, University of Edinburgh, 2006.