GENERATIVE IMPLEMENTATION OF HIERARCHICAL NOVELTY DETECTOR

by

Dagan Eshar

Under the Supervision of
Prof. Daphna Weinshall

A Thesis Submitted in Partial Fulfillment of the
Requirements for the Degree of
Master of Science

at

The School of Computer Science and Engineering
Hebrew University of Jerusalem, Israel 91904

December, 2009

Acknowledgements

I would like to express my gratitude to Prof. Daphna Weinshall, for the guidance, wisdom, knowledge, time spent and all the assistance, throughout all my research. I would like to thank Prof. Bastian Leibe, Andreas Ess and the people of the Computer Vision Lab at ETH, Zurich, for the hospitality and for introducing me to the secrets of their object detector. Special thanks to Alon Zweig for all the efforts, the collaboration and the huge support. Last, thanks to Amit Bleiweiss for the helpful proofreading notes.

Contents

Abstract

In this work I have examined a new approach for novelty identification that was first presented in [59]. This approach was implemented for the task of detecting novel visual object classes, using a state-of-the-art object detection algorithm. The chosen algorithm, denoted CurMethod throughout this work, was developed by Leibe et al. [32] and proved to be successful in various scenarios. In this algorithm, a generative probabilistic method is used to build an implicit model of the learned object class, along with a top down segmentation. The model uses statistics collected only from instances of the target class, and does not utilize any knowledge of background, clutter, or other classes.

A previous implementation of the same approach was recently presented by Zweig in [63]. This implementation, referred as RefMethod later in this work, was based on an essentially different object model, created by Bar-Hillel et. al [2]. This model explicitly represents the object class, using a few highly distinctive parts. It is trained by a hybrid of generative and discriminative techniques, using both positive and negative examples.

While the basic properties of the proposed novelty detection approach were presented in [63], the focus of my work was twofold: reinforcing the validity of the approach by reproducing similar results, using a different object model; and further investigating more subtle aspects of this approach. Note that the choice of the current model and the reference model was done due to their success in classifying a large variety of classes, and their different nature. Indeed, one of the appealing features of this approach is that any model can be similarly used. The results of this work strengthen this last claim.

Chapter 1

Introduction

A number of different methods have been developed to detect and recognize object classes, showing good results when trained on a wide range of publicly available datasets (see e.g. [2, 33, 19]). These algorithms are trained to recognize images of objects from a known class. Ideally, when an object from a new class appears, all existing models should reject it; this is the only indication that a new object class has been seen.

While this approach will work in simple scenarios, it might fail in real-world examples. On the one hand, when no object exists in the image, or when an object is of no interest to the application, a similar indication will be obtained, as if a novel object class was detected. On the other hand, when presented with outliers or noisy images of a known object class , a low recognition likelihood may be achieved from all existing models. Thus, one of the fundamental difficulties with the prevailing novelty detection paradigm, is that negative evidence alone gives rather non-specific evidence, and may be the result of different unrelated reasons.

This work further investigates the task of novel object class identification, i.e. "how to detect an image from a novel class of objects". Unlike the traditional approach to novelty detection, the current method utilizes the natural hierarchy of objects, and develops a more selective constructive approach to novel class identification. The basic observation is that, while a new object should be correctly rejected by all existing models, it might be accepted at some more abstract level of description. For example, a new face should be ideally rejected by all models of known individuals, but accepted by a more abstract model of a 'human face'. This kind of reasoning was implicit in some earlier work on face recognition (see [30]). Relying on positive identification at more abstract levels

of representation allows for subsequent modelling using related classes, as done in [63], where the representation of a new class was built from a learned combination of classifiers at different levels of generality.

Specifically, in Section 2 I describe the new approach for novel class identification. The case I consider is a multi-class scenario, where several class models are already learned. A new sample will be identified as belonging to a new unknown class based on the discrepancy between two classifiers: the first one accepts the new sample and the second rejects it. The two classifiers are hierarchically related, such that the accepting classifier fits a general object class description, while the rejecting classifier fits a more specific object class. Using this approach one can note an important property – a correct detection of a novel class is based on rejection by only a small number of related classes. This allows us to take a discriminative approach to the problem – training the specific classifiers only to distinguish between a particular small group of related subclasses.

To illustrate, suppose that the training data is composed of two object classes of type 'Road Motorbikes' and 'Sport Motorbikes'. One can use all samples from both classes to learn a more abstract notion of the 'Motorbikes' class. When a sample of a new type of bike is presented, such as 'Cross Motorbikes', this sample is similar enough to the known classes, so that it will be accepted as a member of the general level category - 'Motorbikes'. However, specific models which are trained to distinguish 'Road Motorbikes' from 'Sport Motorbikes', and vice versa, any by that are "focused" on the fine details of the class, will probably be able to reject the new sample. This discrepancy is used to indicate that a new sub-class of objects has been encountered, of the general type 'Motorbikes', but of no known specific type.

This approach is general in the sense that it does not depend on the specifications of the underlying object recognition algorithm. To investigate this point, I developed a new implementation of the novelty detector, based on an object model by Leibe et. al [32], and compared it to the existing implementation of Zweig [63]. Both implementations are based on publicly available object recognition methods [2, 32], and are essentially different one from the other. While both methods model object classes based on their appearance and geometrical structure, Bar-Hillel et al. [2] proposed an explicit shape model, learning objects parts and relations among parts, and classifying objects using a maximum-likelihood criterion. Leibe et al. [32], on the other hand, proposed an implicit shape model where a distribution over codebook words and their locations is learned,

and classification is obtained using a generalized Hough transform scheme. Another significant difference is that in [2] single object models are learned using both positive and negative examples. Thus, this is a discriminative algorithm, whereas in [32] only positive examples are used.

I tested these algorithms experimentally (see Section 3.4), using different scenarios: Various levels of discriminativity were developed and tested; Models were tested using images of different qualities, by adding different levels of Gaussian noise; and finally, two types of class hierarchies were used, one tight and one sloppy. Most of the tests were executed on two sets of objects: a facial data set, where the problem is reduced to face verification [43, 12], and a set of motorbike images, extracted from the Caltech256 bench-mark dataset [23].

I found that discriminative methods, which capture distinctions between the related known sub-classes, perform significantly better than generative methods. This is the first evidence that our working hypothesis might be valid, and thus our approach, which allows detecting novel classes using discriminative methods, indeed helps. I also demonstrate in the experiments the importance of modelling the hierarchical relations as tightly as possible, and the specificity of this approach, when confronted with noisy input.

1.1 Thesis Structure

This thesis is organized as follows: The rest of this introductory section reviews related research in the field of visual object recognition, in section 1.2, and novelty detection, in section 1.3. Naturally, I emphasized issues that are relevant to this work, such as the local versus global representation of an object model 1.2.1, the concept of part based models 1.2.2, the usage of object hierarchies to improve recognition 1.2.3, and the differences between discriminative and generative approaches 1.2.4. An overview of existing novelty detection techniques will close this chapter 1.3.

In chapter 2 I present a new approach to novelty detection and its current implementation. I review the basic object models 2.1, the two levels of the novelty classifier (2.2 and 2.3), and the combined algorithm 2.4. Chapter 3 covers the details of the experiments, including datasets, implementation issues, setup, and the results of the experiments (sections 3.4, 3.5, 3.6, and 3.7). Lastly, chapter 4 concludes with a discussion on the results, and possible future research.

1.2 Visual Objects Classification

Visual object classification refers to the task of assigning a correct label to a given image, based on the content of that image. In practice, this usually reduces to either a binary decision ("does the target object appear in the image?"), a multiple choice ("which of the target objects is most likely to appear in the image?"), or a detection task ("where is the target object in the image?").

The framework usually includes a description of the object or class of objects, a classification method that distinguishes the target object from the other objects, and a recognition method that will tell which is the most likely class to appear in a new unseen image. There is a vast amount of different algorithms that try to accomplish one or more of the above tasks, and new ones still appear. I will briefly review the previous work conducted in the field, emphasizing the distinctions that are relevant to this work.

1.2.1 Local vs. Global Features

One of the traditional disputations in the field of object classification was the usage of local features against using global features. In the first category, the common approach is known as "bag of words"[1]. Here, many interesting visual features are collected from all over the image, and based on the statistics of these features, a decision is made (either the target object is present or not). Each of these features represents only a small area of the image, and does not contain information about the global shape of the object. In contrast, the latter category includes features that capture the appearance of the object as a whole, such as template-based and shape-based methods. While "bag of words" methods are computationally cheap, robust to occlusions, and invariant to different transformations (depending on the characteristics of the descriptor), they lack the accuracy and discriminability of shape-based methods.

When talking about **local features** we refer to an unordered set of locations in the image (called also "interest points"), which are represented by some vector of properties (usually called the feature's "description" or "descriptor"). These features usually do not contain any information about the shape or geometry of the object, and their description refers only to a small area of pixels (patch) around the interest point. Once a set of

[1] The term "bag of words" comes from text analysis tools, where the statistics were actually calculated over the distribution of the words in the text

feature-descriptors is collected, a histogram of these features is calculated, and used as the image representation. This representation is used in order to learn the target object's model, at the training stage, and in order to classify a new image. Various classification methods can be applied at this stage, among them Nearest Neighbor techniques, SVM, Gaussian Mixture Models, and others (see for example [14, 36, 60]).

The methods for finding the interest points (detectors) and describing them (descriptors) are varied and go back at least to Beaudet's Hessian corner detector in the 1970's (see [46]). The goals, however, remain the same to this day: quickly and efficiently detect those locations in the image which hold the most significant information and yield the best representation, and describe them in a way that achieves uniqueness and repeatability. This means, in practice, that different locations over an object should have different descriptions, while the same location over different instances of the same object should have a similar description. Note that the appearance of different instances of an object or a class of objects can change due to variations in point of view, scale, illumination, partial occlusion, non-rigid deformations, noise, and inter-class variability.

The Harris Corner Detector (reviewed in [54]) was one of the first successful attempts to implement such an interest point detector, and perhaps the most famous even to this day. Both Tuytelaars and Mikolajczyk [54], and Moreels and Perona [46], survey many other approaches, among them: the similar Hessian corner detector; the somehow complementary blob detectors, such as Maximally Stable Extremal Regions (MSER); Intensity-Based Regions (IBR) and Edge-Based Regions (EBR); scale preserving variants using Laplacian Pyramids or Difference of Gaussians (DoG) (e.g. Harris-Laplace); segmentation based methods (using Graph-Cut or similar algorithms); detectors based on machine-learning algorithms (SVM, Decision Trees); and methods finding salient regions in the image, such as the detector by Kadir and Brady, based on entropy and self-dissimilarities [28].

When talking about local feature descriptors, our goal is to develop a vector representation of the interest point, that will specific to the described location, on one hand, but invariant to different transformations, on the other hand. The most intuitive way is simply to describe a fixed size patch around the interest point, usually after applying color or intensity normalization techniques. This can be done by simply converting the image patch to a vector, but this lacks almost any invariance characteristics. One solution is using PCA, FFT, DCT, or similar algebraic dimension reduction, in order to

improve the invariance to noise and allow better generalisation (see for example [3]).

In order to introduce invariance to scale and rotations, Lowe developed SIFT (Scale Invariant Feature Transform) descriptors [36]. This method contains a normalized histogram of gradients around the point, for scales that exhibit extrema in the DoG of the scale-space. This approach was followed by many variants as PCA-SIFT [29] and the more efficient SURF (Speeded Up Robust Features)[24]. For region based detectors, a natural description is simple blobs, such as ellipses for the previously mentioned MSER detector [42]. Finally, the Shape Contexts descriptor, which was introduced by Belongie, improves shape specificity, by representing a histogram of locations and orientations of neighbouring edges [6].

In contrast to the local features approach, researchers developed many different methods for **global features**. This description tries to capture the essence of the image/object/class as a whole, instead of describing many small locations. Global descriptors can be very accurate and specific, but usually lack the robustness and invariance of the local features approach.

Among these techniques I will first mention histogram-based methods, which describe the general properties of the image, as color, edge-orientations, oriented gradients, or responses to different filters (Haar-like, Gabor, etc) [11, 1]. Template matching, in contrast, emphasizes on the global shape, as was efficiently implemented for a pyramid of edges in the Chamfer Matching [8]. Eigenfaces (or eigenvectors in general), and Fisherfaces apply factorization and representation of the learned object into a lower-dimension algebraic base, thus capturing the prominent features only (see a comparison in [5]). The Active Shape Models solve the optimal representation of the object's contour, and the Active Appearance Models combine that with the appearance/texture (see [13]). On the other hand, already in 1978 Marr and Nishihara [40] have recognized the advantages of sticks representation (or skeletoning, as sometimes called today) for representation. A few years ago Viola and Jones described in [57] a robust real-time detector, which combined shape and appearance. They used a boosting scheme to learn a cascade of simple Haar features, resulting in a very efficient classifier.

1.2.2 Part Based Models

Since these two approaches are so different, they have complementary strengths and weaknesses. It is not surprising that the latest developments in the field try to combine

them and thus compensate for one another's limitations. This approach is often referred to as "**part based models**" or "**constellation models**", indicating that we have (at least) two different levels of description: a general one, describing the appearance and structure of the whole object and the spatial relations between its smaller parts, and a part which describes a specific location on the object. Note that this can be a multi-level system, with a hierarchy of parts, as proposed by Marr and Nishihara [40]. Think for example about a possible description of a person, including the following parts and model: the *body* at the center, the *head* on top, two *hands* at the top-left and top-right sides, two *feet* at the bottom-left and bottom-right sides. We can further describe the *hand* part as: *arm* on the top, *hand* in the middle, and *palm* at the bottom. Similarly, we can describe the *palm* as a composition of *fingers* and so on.

On the one hand, this approach is more accurate and specific than the "bag of words" or "global representation" described above, since it models the shape of the objects in addition to its appearance. On the other hand, it can represent classes with relatively large variability, as it allows some flexibility in the shape, as long as similar parts are present in a similar formation. The main disadvantage with this models is that they tend to be complicated, with many parameters to estimate and tune. This makes the methods computationally expensive, both in the learning and in the recognition stages, and requires additional manual human interference (e.g. for labelling images and tuning parameters). Specifically, the computational cost might grow exponentially in the number of parts, which limits most of the models to 5-10 components.

Early works appeared already in 1973, when Fischler and Elschlager created a physically-based model, in which the parts (components) are described as masses connected by springs [21]. This work was later revisited and implemented efficiently by Felzenszwalb and Huttenlocher [17], who showed how to solve the minimization of an energy function, measuring both a match cost for each part and a deformation cost for each pair of connected parts. Fergus et al. present a probabilistic approach, modeling objects as random constellations of parts, where shape, appearance, occlusion and scale are learned statistically [18].

Recently, Star-shaped constellations have become popular due to their reduced computational cost, resulting in many works, such as Bar-Hillel [2] and Fergus [19] (explicitly learning a small number of prominent parts), or Leibe [32] (implicitly learning the shape of the object using thousands of simple features). The models of Bar-Hillel and Leibe

will be described in more detail in section 2.1. Lately, Osadchy and Morash showed in [48] that even a loose shape model can gain superior performance. In their model, image parts are represented by an ordered set of bags-of-features, computed in local overlapping areas around the part, and a semantic distance between images is obtained by matching parts using both their appearance and their context distribution.

A different approach was taken by Tomaso Poggio's group [50], and by [26], who simulate visual cortex mechanisms. Here we have a multi-layers model with hierarchical features, where each feature is a combination of features from lower layers (except the lowest, first, layer). The hierarchy is created so that each layer is more invariant to transformations than the previous one, and more specific (detailed). In Riesenhuber et al. the model is biologically based, inspired by the early stages of the visual cortex and tuned according to researches on humans and animals. The classification is done using standard off-the-shelf tools at an intermediate level representation. In [26] the layers get more and more specific and complex, until the uppermost layer contains the classification result for a target object.

1.2.3 Hierarchical Classifiers

As the algorithms for classification rapidly improve, new goals are set by researchers in order to solve real world problems. One of the common tasks is to identify a very large number of classes, from as few as possible training examples. The new challenges now, in addition to the accuracy of recognition, are: limited computation resources (CPU, memory, time); very large amounts of data, but limited or inaccurate training information (annotated datasets); and the very large variability of the classes. This scenario implies that we might have many object classes with only a few training samples for each class. Thus we have to transfer knowledge between classes and take advantage of similarities between related objects. The development of hierarchical classifiers is one of the promising directions for solving this problem.

Fei-Fei [16] uses prior probabilities learned from three classes in order to improve the classifier for a new class, in a one-shot-learning scenario (only 1-6 positive examples). Bart and Ullman [4] learn a novel class by selecting features that proved useful for previously learned classification tasks, and adapting these features to the new classification task. They show that when ten classes were already learned, even a single novel-class example is enough for learning useful classification features. Fink [20] learns a metric be-

tween instances of object classes, that emphasizes inter-class variability over intra-class variability, and then he applies a nearest neighbour classifier to distinguish between the learned classes. With this approach he succeeds to classify novel classes using a single training example.

Sudderth et al.[52] further define three different levels of information sharing in their model for similar scenarios: first, parts are composed by spatially combining low level features from a single pool; second, objects are defined by histograms of parts, and third, scenes are classified using spatial relationships between those objects. Levi et al. [34] assume that similar features are useful to classify sibling categories in a hierarchy (e.g. fruits), and apply a boosting framework to select those features, based only on few examples per class.

Bar-Hillel and Weinshall take a different approach for exploiting class similarities in a hierarchy, showing that the combination of classifiers from two different levels of abstraction, results in a better recognition performance [3]. A more general (abstract) classifier gives good recall, while a specific one has higher precision, thus combining them wisely gains from both worlds. This last property is exploited also in the proposed novelty detector.

A different way to exploit similarities between classes is demonstrated by Marszalek et al. [41], who take advantage of semantic hierarchical relations from a language thesaurus to build a multiple layers classifier. Detection of objects in an image is done using a top-down search of the tree, where in each stage a discriminative classifier is trained to separate all semantic concepts belonging to one node from all the concepts belonging to its siblings (in a similar fashion to[63]).

All the above works assume that the hierarchy of objects is given, at least to some extent. Recently there has been some interest in learning the object hierarchies in order to remove this assumption [51]. Such results can be naturally integrated in the current framework, but it is beyond the scope of this work.

1.2.4 Discriminative vs. Generative Approaches

Another important distinction between object classifiers is the question of discriminative vs. generative models. Since one of the goals of this work is to compare a generative approach [32] to a discriminative one [2] in a similar framework [63], we will discuss some of the main issues regarding this question.

Intuitively when talking about a *discriminative* method we refer to a classifier that *discriminates* the different object classes (often it will be the target class and a background class), while a *generative* method defines a model that explains how to *generate* an instance of the target object class. Adopting Bishop's definitions [7], from a probabilistic perspective, our goal is to find the conditional distribution p(c|x), where x is the representation of the data sample, and c is the class label.

Discriminative Methods One common approach to this problem is to represent the conditional distribution using a parametric model, and then to determine the parameters using a training set, consisting of pairs $\{x_n; c_n\}$ of input vectors along with their corresponding target output vectors. The resulting conditional distribution can be used to make predictions of c for new values of x. This is known as a discriminative approach, since the conditional distribution discriminates directly between the different values of c.

As already mentioned, there are numerous classification algorithms, and all fall in one of these categories, or use a combination of them. Some of the popular discriminative methods are: Logistic Regression, Support Vector Machines (SVM), traditional Neural Networks , and Decision Trees (see reviews in [27, 31]).

Generative Methods An alternative approach is to find the joint distribution p(x,c), expressed for instance as a parametric model, and then subsequently use this joint distribution to evaluate the conditional p(c|x) in order to make predictions of c for new values of x. This is known as a generative approach since by sampling from the joint distribution it is possible to generate synthetic examples of the feature vector x.

Some of the popular generative methods are: Naïve Bayes, Gaussians, Mixtures of Gaussians, Mixtures of Multinomials, Mixtures of Experts, Hidden Markov Models, Sigmoidal Belief Networks, Bayesian Networks, and Markov Random Fields [27].

Recently researchers started to investigate the strengths and weaknesses of the two approaches in general, not only for specific classifiers. There are at least four benefits for using discriminative approaches over generative ones. First, it was shown that the asymptotic error of discriminative methods (in the number of training examples) is lower than for generative ones, when using simple learning models [47]. Second, in order to use a generative method, one must come up with a relevant model, which can be hard

to define, and task dependant. This classifier can have poor generalization if the learned model is inaccurate [55, 10]. Specifically it may be hard to estimate well the prior probabilities of the parameters [31].

A third reason to prefer discriminative methods is that generative models do not take advantage of available "negative" examples (the other classes). Thus the learned model might not be optimal for the discrimination task, resulting in inferior performance. This is especially true when the target classes are similar (low between-class variability)[2]. Last, generative classifiers tend to be more complex, and thus more computationally expensive, since it is required to optimize the model [31].

On the other hand, there are at least five good reasons to choose generative methods. First, when there are few training examples, generative methods achieve better results [47, 55]. Second, in [7] it is shown that generative methods can improve classification by using unlabelled training data, while this is not possible with pure discriminative methods. Third, prior information, which can be naturally incorporated in generative methods, was shown to be useful in few-training-samples scenario [31]. It is not clear how to use this knowledge in discriminative methods. Fourth, when learning to identify a single class, discriminative approaches must get as input both positive and negative examples, in order to learn a separation. As a consequence we have to collect data containing all the variability of the non-class examples, which can be a very difficult task in real world applications [25].

The last reason arises when building systems that learn thousands of classes. In this case it might not be feasible to learn all classes at once, as required by discriminative methods [31]. Moreover, when employing discriminative approaches in the multiple classes scenario, it is not clear how to detect the most likely class: one-versus-all classifiers might be less efficient (as they are less discriminative), while one-versus-one classifiers require an exponential number of classifiers (since we need to compare every two classes). Similarly, a difficulty arises when adding new classes to an existing discriminative system, as this might force the retraining of all existing classifiers. In the generative approach, on the other hand, each classifier can be trained independently, and if needed, even different models can be used for different classes.

[2]Experiments proving this intuitive claim are presented here in section 3.5

Hybrid Methods Because of the complementary nature of the above strengths and weaknesses, researchers believe that we can get the best of both worlds by combining them. I will explore recently proposed algorithms for hybrid methods, showing representative examples for each one. A popular approach, due to the success of bag-of-words methods, was the creation of discriminative codebooks, which was implemented in several ways, including [62, 58, 45, 60, 56]. These codebooks can be naturally used by generative algorithms. Yang et al. [35] went a step beyond discriminative codebooks, defining a unified framework for optimizing the codebook specifically for the learned classifier (a discriminative one in their case).

The converse approach, learning a discriminative classifier over generative features or models, was shown by [25, 9, 26]. Holub et al. [25] use Fisher Scores (Fisher Kernels) extracted from a constellation model; Bosch [9] performs SVM classification over a pLSA model; and Huang and LeCun [26] present a hybrid system, where a convolutional network represents the images, and an SVM is trained for the classification . A two stages classifier was described in [22], where hypotheses for detections are first generated by ISM model and then refined using SVM.

Other researchers developed hybrid optimization problems, which introduce families of classifiers by interpolating the two approaches. Works, such as Bouchard and Triggs [10], or Bishop and Lasserre [7], claim that balancing the effects of the generative method with the variance of the discriminative one, outperforms both approaches, and they investigate the definitions of such scenarios.

1.3 Novelty Detection

The proposed implementation uses a recently presented approach to novelty detection, based on the detection of incongruities [59]. Other existing methods for novelty detection, in contrast, identify novel events by estimating the boundaries of the known classes, in a generative manner (see [38, 39] for recent reviews). For example, some estimate a spherically shaped boundary around a single class data set [53], others learn a hyper-plane which separates the class representation from the rest of the feature space (one class SVM) [49], and others still utilize the non parametric Parzen-window density estimation approach [61].

A few methods use a multi-class discriminative approach, as in the case of [15] for the

detection of novel objects in videos, and [12] for the specific task of face verification. To our knowledge, all novelty detection approaches which do not rely on samples of outliers or otherwise model the outliers' distribution, detect novelty by rejecting normality (i.e. novelty is identified when the classifiers of all the known classes reject a new sample). One of the main problems with this approach is that when some input is rejected, we do not have any information about the reason for the rejection. Specifically we cannot distinguish an unknown (but interesting) event from noise. The current approach addresses this problem directly, by identifying novelty only in events that satisfy certain conditions (in other words, that are accepted by some general level classifier as "interesting").

Chapter 2

Novelty Detection Approach & Algorithm

In order to identify novel classes, the proposed approach is to detect discrepancies between two levels of classifiers that are hierarchically related. The first level, explained in section 2.2, consists of a single 'general category' classifier, which is trained to recognize instances of any known sub-class. The second level is based on a set of classifiers, each trained to make more specific distinctions and classify objects from a small group of related known sub-classes. Using these specific classifiers, a single classifier is built, which recognizes a new sample as either belonging to one of the set of known sub-classes or not. Further details are given in section 2.3.

Finally we look for a discrepancy between the inferences made by the two final classifiers, from the two different levels. Specifically we are interested in the case, where the general-level classifier accepts a given input, while the specific-level classifier rejects it. This indicates that the new sample belongs to the general category but not to any specific sub-class, or in other words, it is a novel, interesting class. In section 2.4 I elaborate on the combination of the classifiers. This algorithm is described more formally in 2.3.1.

Note that even though I presented a two-levels algorithm, the approach can be easily extended to more levels, where each intermediate level serves as the 'specific level' for the parent level, and as the 'general level' for the child levels. Developing this extended hierarchy is out of the scope of this work.

It is interesting to see that the classifiers at the different levels can be chosen in-

dependently, and it is an open question how to select the best classifiers for the tasks. Later in this work I will further investigate this issue. Section 2.1 shows the classification algorithm that was chosen for the tests, and the reference algorithm, by Zweig [63]. Both algorithms can learn any visual object class, and both proved to be comparable to the state of the art algorithms.

2.1 Basic Object Classifiers Implementation

In order to validate the above mentioned approach for novelty detection, I have implemented a classifier, based on a well known object detector (see CurMethod in section 2.1.1). This implementation was tested and compared to an existing implementation, which uses a different object classifier (see RefMethod in 2.1.2). Each implementation has its own strengths and weaknesses. As I will now show, both of the methods belong to the part-based-models family, and both use a star-shaped constellation of parts. These parts, in turn, are chosen from a set of features that was extracted by an interest point detector, and represented by a chosen descriptor.

On the other hand, the current method is a purely generative method, with thousands of less significant features, while the previous one uses a combination of generative representation with a few discriminatively selected features. Due to their nature, the former classifier uses only positive examples, while the latter needs both positive and negative examples. We should also note that RefMethod was designed to discriminate between two target classes, while CurMethod was designed to locate multiple instances of a single class. Thus, both methods had to be slightly modified, in order to suit the task of image classification.

A common preprocessing step is applied to all input images with both methods, at the training and at the classification phases. Each image is first processed by an interest point detector, resulting in an unordered set of visual features. This feature set usually contains several hundred features in various scales, with considerable overlap. Each feature is represented by a visual descriptor, and this description vector is augmented by the location of the feature and its scale. The final feature descriptors are the image representation (i.e. the input) for the methods below.

2.1.1 CurMethod

The chosen object detection algorithm, in [32, 33], starts by extracting a codebook of local appearance descriptors. This codebook is based on the detection of interest points in the training images of the learned class. Next, for each codeword, a vector of possible occurrences over the instances of the class is calculated. Each entry contains the ID of the codeword, its location (relative to the center of the object), its scale, and a probability weight. This weight is proportional to the frequency of appearances of the codeword in this specific location and scale. If available in the training dataset, a segmentation mask (foreground/background probability per pixel) is also kept for each occurrence. At the end of this stage we obtain a vector of occurrences per class, indicating the probability to see each feature at a specific location and scale, relative to the center of the object. Notice that in contrast to many popular constellation models, there could be hundreds or thousands of occurrences for a single object instance (compare for example to the number of parts selected in RefMethod).

At the detection stage, the first step is searching all codebook features at all possible locations in the image. Then, using the Generalized Hough Transform, each codeword occurrence gives a weighted vote for one or more hypotheses (location and scale of a target class instance). Local maxima are searched in this 3D space (X-coordinate * Y-coordinate * Scale) in order to decrease the number of plausible hypotheses. For each hypothesis a score is now calculated by summing its votes, and only those hypotheses, which passed a learned threshold, are kept. An MDL score is calculated for the final hypotheses, by optimizing the percentage of the image that is explained by the hypothesis. This is done by applying the segmentation masks that are attached to the occurrences, and calculating the amount of pixels that are covered by the foreground part of the masks.

The final score for an input image is obtained by choosing the score of the most likely hypothesis and comparing it to a learned threshold. Note, that unlike RefMethod, this method is based solely on positive examples during the entire learning phase, and is not trained discriminatively.

2.1.2 RefMethod

The reference algorithm from [2] learns a generative relational part-based object model with P parts, where each part is implemented by a single visual feature. The model is described by the appearance of the parts and their location and scale, relative to some object center. These relations are captured by a star-like Bayesian network. The model parameters are discriminatively optimized using an extended boosting process, which uses both positive (object) and negative (background) input images. Based on this model and some simplifying assumptions, the likelihood ratio test function is approximated (using the MAP interpretation of the model) by

$$F(\mathbf{x}) = \max_{C} \sum_{k=1}^{P} \max_{u \in Q(\mathbf{x})} \log p(u|C, \theta^k) - \nu \tag{2.1}$$

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

In order to allow single class classification, I train the classifier with positive examples of the target object class and negative examples of "other" instances. These can be clutter images, unrelated objects, other classes from the hierarchy, or sibling classes in the hierarchy, depending on the task. A threshold is learned in the training phase, to distinguish the target class from the other classes.

2.2 General Category Level Classifier

In order to learn the general category level classifier, we consider a small set of related known classes as being instances of a single (higher level, or more abstract) class. In accordance, all the examples from these known classes are regarded as the positive set of training examples for the more abstract class. For example, both Road and Sport motorcycles are examples of the general category Motorcycles. When a negative set of examples is needed for training (as in RefMethod), we use either clutter or different unrelated objects (none of which is from the known siblings). As will be shown in section 3.4, this general classifier demonstrates high acceptance rates when tested on the novel sub-classes.

Let us note that we cannot assume in advance that there exists any similarity of

appearance or shape between the general categories. This is essentially task dependant, where in the worst case there might be hundreds of categories, which might be extremely different one from the other (e.g. motorcycles, cars, people, fish, quadrupeds, tables, and so on). For these reasons different general category classifiers may use different representations, classification methods, and scores, and for such we cannot compare them. Therefore we treat each category independently.

When given an input image we run the general category classifier; if the score is above a learned threshold we accept the image, otherwise we reject it, regardless the decision of other general level classifiers.

2.3 Specific Category Level Classifier

At the specific level classifier the problem is essentially reduced to the standard novelty detection task of deciding whether a new sample belongs to any of the known classes or to an unknown class. However, the situation is somewhat unique and we take advantage of this: while there are multiple known classes, we are not interested in comparing all of them. We only compare a limited number of sibling classes under a single hierarchy branch. Here all the classes are part of one general category (in practice the number is bounded by the degree of the hierarchical tree).

Having a common parent, these classes share some shape and appearance properties, leading us to the assumption that a model based representation might be useful. Furthermore, under these conditions, a generative classifier may have a hard time representing the unique features of each class, while a discriminative approach could take advantage of that by ignoring the common features and emphasizing the discriminating ones. Following the example of the previous section, at this level we know that we are looking at a picture of a motorcycle, we now need to decide if it is a known type (Sport or Road), or a new one.

The training procedure of the specific level classifier is summarized below in 2.3.1 with details following, while the classification process is described in 2.4.

2.3.1 Training the Specific Category Classier

Algorithm 1:

1. For each specific class, build a discriminative classifier with:
   positive examples: all images from the specific class
   negative examples: images from all sibling classes

2. Compute the Normalized Confidence function.

3. Choose a classification threshold for novel classes.

Step 1: Training the single class classifiers

To solve the multi-class classification problem, in Step 1 of the algorithm I train a discriminative object class classifier for each of the known specific classes and classify a new sample according to the most likely classification (max decision). For each of the two implemented object recognition methods used, we exploit the discriminative information in a different manner.

CurMethod uses only positive examples when building the object class model. Given such a model, object class classification is done by comparing the score of a new image to a threshold, see section 2.1.1. In order to incorporate the discriminative information of sibling subclasses, I choose the threshold that achieves an equal error rate for the classification of the known class against its known siblings on a validation set, for each single class classifier.

Using RefMethod, it is possible to incorporate the discriminative information in the training phase of each of the single known classes-specific classifiers. Each single class classifier is trained using all images from its siblings (other known classes under the same general level class) as negative examples. Thus, the specific level object model learned for each known class is optimized to separate the class from its siblings.

Step 2: Normalized Confidence Score

For a new sample $\mathbf{x}$ which is classified as C_i, we obtain an estimate for classification confidence $V_{C_i}(\mathbf{x})$. This value is either the output of the learned classifier, when using RefMethod, or the distance to the learned threshold, when using CurMethod. After the max operation in Step 1, this value reflects the classification confidence of the multi-class classifier. Given this estimate, the goal is to derive a more accurate measure of

confidence in order to decide whether or not the classified sample belongs to the group of known sub-classes.

To do this, a normalized score function is defined. This score normalizes the confidence estimate $V_{C_i}(\mathbf{x})$ relative to the confidence estimates of correct classifications and wrong classifications, which was measured during training or validation.

Specifically, let $V^c_{C_i}$ denote the average confidence of train or validation examples, classified correctly as C_i. Let $V^w_{C_i}$ denote the average confidence of train or validation examples from all other sub-classes, mistakenly classified as belonging to class C_i. The normalized score $S(\mathbf{x})$ of $\mathbf{x}$ is calculated as follows:

$$S(\mathbf{x}) = \frac{(V_{C_i}(\mathbf{x}) - V^w_{C_i})}{(V^c_{C_i} - V^w_{C_i})} \tag{2.2}$$

If the classes can be well separated during training, that is $V^c_{C_i} >> V^w_{C_i}$ and both groups have low variance, the normalized score provides a reliable confidence measure for the multi-class classification.

Step 3: Choosing a novelty detection threshold

Unlike the typical discriminative learning scenario, where positive and negative examples are given during training, in the case of novelty detection no actual negative examples are known during training. In particular, when the specific-object classifiers are trained, they are given no example of the unknown sub-classes whose detection is the goal of the whole process. Moreover, note that the discriminated classes can be very similar, as they belong to the same general category. Following our example, it is now required to distinguish a Sport Motorbike from a Road Motorbike, and not only a Motorbike from other objects.

Under these conditions it becomes advantageous to set rather conservative limits on the learned classifiers, more than indicated by the train set. In other words, in order to classify a new sample as known (a positive example for the final classifier), a higher classification confidence will be needed. This is done by setting the threshold of the normalized confidence measure to reject more than originally intended during training. As can be deducted from step 2, the normalized confidence measure lies in the range [0..1], thus we set the threshold in our experiments to 0.5.

	SL Accepts	**SL Rejects**
GL Accepts	Known sub-class	Unknown sub-class
GL Rejects	Irrelevant event	

Table 2.1: Basic classification results of the novelty detector

2.4 Combined Novelty Classifier

Putting it all together we get the following novelty classifier:

Algorithm 2:

1. Given a new image, set it as input to all the general category classifiers
   - If no classifier accepts the image, tag it as "irrelevant" (or uninteresting) and quit.
   - If any classifier accepts the image, mark it as AGC (accepting general category) and proceed to step 2.

2. for each AGC perform:

   (a) Set the image as input to all the specific category classifiers that are children of AGC.

   (b) Pick the classifier that gained the maximal normalized-confidence-score and mark it MSC (maximal specific category).

   (c) Compare the score of MSC to the novelty-detection-threshold:
   - If the score exceeds the threshold, tag the image as an instance of the MSC
   - Otherwise tag the image as an instance of a new unknown subclass of AGC

If we look at a single general category, there are basically two stages: running the general category classifier, and running the specific categories classifiers. One way to analyze this is as two random variables, GL (general level classifier) and SL (specific level classifiers). Both variables will either have the value "accept" or "reject". Note that for SL, "accept" means that any of the sub-classes classifiers accepted, and "reject" happens only if all sub-classes rejected. The result is the table 2.1, containing the responses of the classifiers and their interpretations.

	SL Accepts	**SL Rejects**
GL Accepts	Known sub-class	Unknown sub-class
GL Rejects	Misclassification?	Irrelevant event

Table 2.2: Extended classification results of the novelty detector

The bottom line of table 2.1 is actually a single option, since SL is not running in this scenario. We could modify the algorithm to run independently the two stages, and then we get the responses in table 2.2.

By design, the bottom-left response in table 2.2 should never occur, since all SL classifiers were trained on a data, which is partial to the GL classifier. Thus, this output may be important as indication to a classification error, and this might be useful when evaluating the system. Alternatively, if this response is consistent across multiple examples, it probably indicates something more fundamental. Remember that the GL and the SL classifiers can be essentially different, and rely on different aspects of the input. Thus, if the misclassified examples are really instance of SL, this may suggest a problem with the hierarchy, specifically that the GL is not general enough. The bottom line is that the bottom-left square is tricky, and although it should put a warning sign in front of our face, this warning should be handled with real care.

Chapter 3

Experiments

3.1 Datasets

As in [63] I used two different hierarchies in my experiments, as reviewed below.

In the *Faces* hierarchy, the general (parent) category level contains images of frontal female faces, taken from [37]. The specific (offspring) classes are six different individuals from this source (see Figure 3.1). A mixture of general object images was used as negative examples.

The *Motorbikes* hierarchy uses, as the general category, images from Caltech-256 [23] that are tagged as *Motorbike*. This category was manually split into the following specific classes: 'Sport-Motorbikes', 'Road-Motorbikes' and 'Cross-Motorbikes' (see Figure 3.2). 22 other object categories were also taken from [23], in order to serve together with the Motorbikes as the pool of object classes. These were used both as unseen-objects (described in section 3.4) and for the random grouping (described in section 3.7). The 'Clutter' category images were used as negative examples. Figure 3.7 shows the Motorbikes hierarchy and a possible hierarchy of some unseen objects.

The two datasets exhibit different challenges, resulting from different variance among the hierarchy classes. In the Faces dataset, the specific classes have a very low variance (same size, pose, and lighting), and similarly the general category. On the other hand, the Motorbikes dataset has a relatively high within-class variance both in the specific and the general categories, and a low between-class variance.

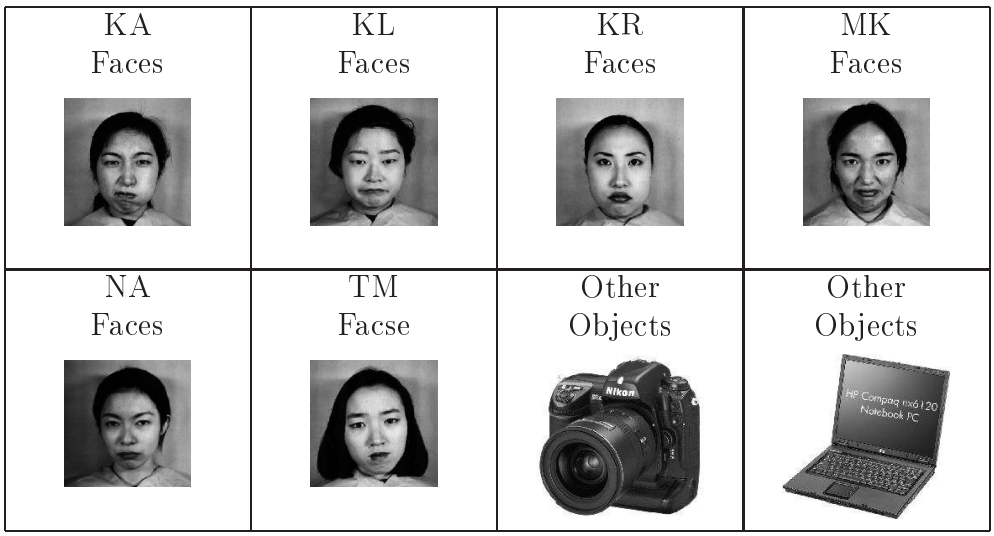


Figure 3.1: Examples from the object classes in the 'Faces' hierarchy, taken from [37]. The specific classes represent different individuals. A mixture of general object images were used as negative examples.

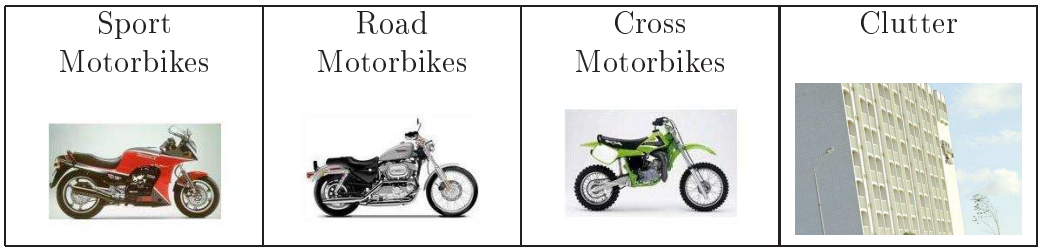


Figure 3.2: Examples from the object classes and clutter images, used for the 'Motorbikes' hierarchy. The specific offspring levels are: 'Sport-Motorbikes', 'Road-Motorbikes' and 'Cross-Motorbikes'. These images were taken from the Caltech-256 [23] dataset. Images of the Clutter category are used as negative examples.

3.2 Models Representation

CurMethod was trained using the Harris-Laplace feature detector [44], and the features where represented using the Shape-Context descriptor [6]. The clustering of the visual codebook was tuned to produce codebooks of several hundreds visual words, resulting in occurrences vectors holding tens thousands of entries.

RefMethod was trained using the Kadir & Brady feature detector [28]. The features are normalized to uniform size, zero mean and unit variance. They are then represented

using their first 15 DCT coefficients. The model contains 30 parts.

It is important to note that since the role of the visual words differs significantly in the two methods, I did not try to use the same feature detectors and descriptors as in RefMethod. This might have given an undesired advantage to one method over the other. Instead, I decided to use the configurations that gave the best results for each method independently, and focus on comparing only the classification results[1].

3.3 Experiments Setup

A single experiment repetition consisted the following steps (as explained in 2.3.1):

- Random splitting of the images from the specific-classes to train and test sets
- Training all known specific level classifiers, each one using its own train set
- For each known class c:
  - Selecting c as the unknown class, and the other specific level classes as the known classes
  - Training the general level classifier using all the known-classes train set
  - Learning the thresholds for each classifier
  - Running all the classifiers on the test set, and evaluating the classification results

All experiments were repeated at least 25 times with different random sampling of test and train images. In each repetition, for each dataset with n classes, n conditions are simulated, leaving each of the classes out as the unknown (novel) class, and using the rest as the known classes. 39 images were used for the training of each specific level class in the 'Motorbikes' hierarchy, and 15 images in the 'Faces' hierarchy.

[1] For a more comprehensive discussion on the issue of comparing generative and discriminative methods in a fair framework, see section 4

3.4 Basic Results

Figure 3.3 shows classification results for the discriminative approach described in Section 2.1.2. These results show the classification rates for the different types of test samples:

- *Known* - samples of all classes that were used at the training phase
- *Unknown* - samples from the novel class that belongs to the same General level as the Known classes, but has been left out during training
- *Background* - samples of classes that do not belong to the general level, and were used as negative examples for the threshold computation (in CurMethod), or during the training phase of the General level classifier (in RefMethod)
- *Unseen* - samples of objects from new classes that were not seen during the training phase, neither as positive nor as negative examples

The three possible types of classification are *Known*, *Unknown*, and *Background*. These match the three different results explained in table 2.1 , with the exception that the Background option is referred as *Irrelevant* in the table[2].

Note that a sample is classified as *Known* if it is accepted by classifiers in both levels, meaning that it belongs to one of the known (learned) classes (top-left cell of the table). The *Unknown* classification indicates a sample that was accepted by the general level classifier, but rejected by all the specific classifiers, meaning that it is a novel sub-class of a known general category (top-right cell of the table). Last, *Background* samples are those that were rejected by the general level classifier, and thus are not instances of (any) general category (bottom row of the table)[3].

The results in Fig. 3.3 show the desired effects: each set of samples - Known, Unknown and Background, has the highest rate of correct classification for its own category.

As desired, we also see similar recognition rates (or high acceptance rates) of the Known and Unknown classes by the general level classifier, indicating that both are

[2]The different naming is due to conventions in the visual object classification field, where the natural classification distinguishes objects from background

[3]In order to add the fourth classification, Unseen, another level must be added to the hierarchy. This will include a higher level classifier, trained using all seen objects as positive samples and all non-objects as negative samples, and specific classifiers for each background object. This task was not in the scope of the current work.

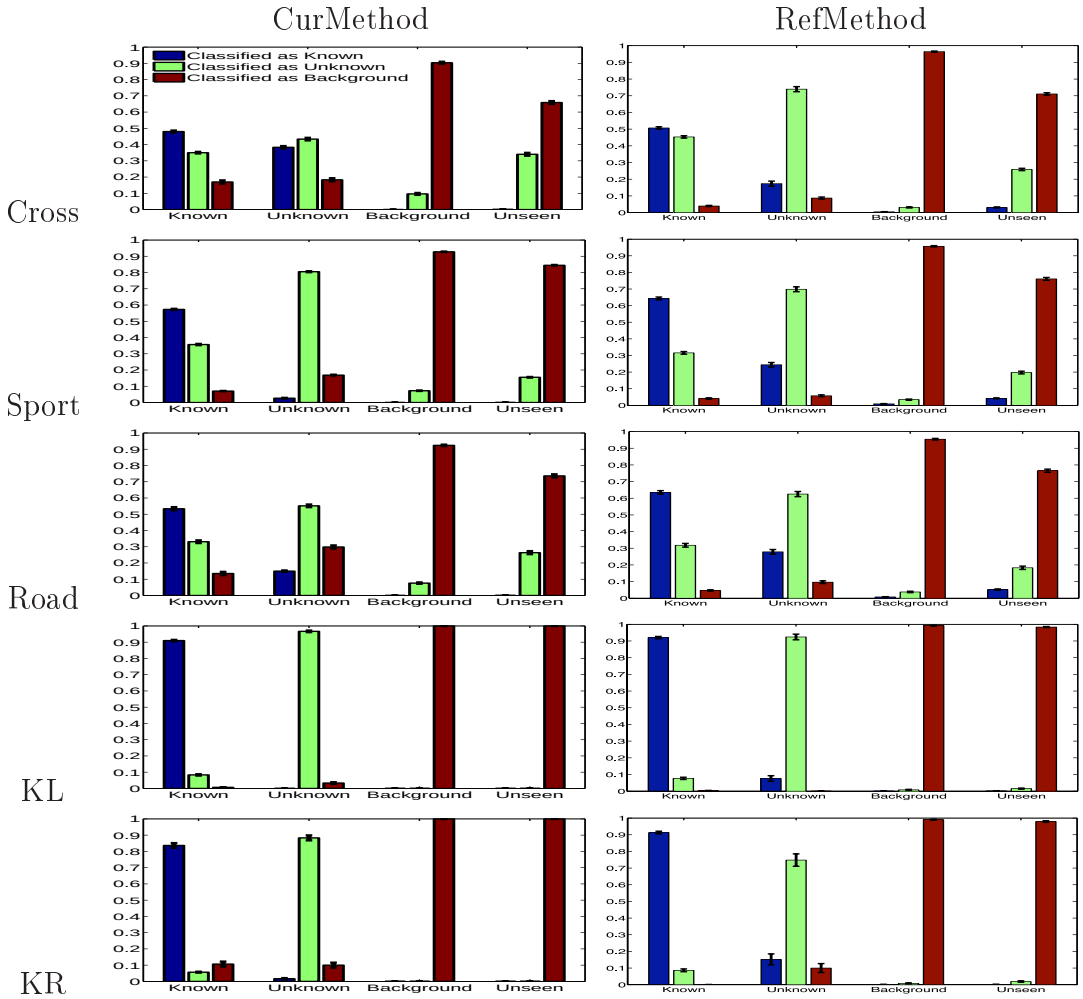


Figure 3.3: Classification ratios for different sample types. Each plot contains 4 groups of bars, representing the sample type. From left to right the groups are: Known Classes, Unknown Class, Background Images, and Unseen Objects. The classification ratios correspond to the 3 different bars: Known (left), Unknown (middle), and Background (right). Each row displays a single class that was left out as Unknown. Namely: Cross, Sport, and Road (from the Motorbikes hierarchy), KL, and KR (from the Faces ierarchy). The left column shows the results of CurMethod. Results for RefMethod are presented in the right column for comparison. Details are given in section 3.4

regarded as belonging to the same general level. This fact can be seen by the low percentage of classification as Background (rightmost bar in each category). Nevertheless, for most of the classes (except class KR) we can see a slightly better acceptance ratio for the Known classes (leftmost group in each plot). This is not surprising if we remember

that no sample of the Unknown class was used during training of the general classifier.

Finally, samples from the Unseen set are rejected correctly by the general level classifier. Again, it can be noted that the classification ratio of Unseen as Unknown is higher than the classification of Background as unknown. This can be attributed to the fact that no Unseen samples were used for training.

Comparing the results to those of RefMethod[4], we can see a few interesting effects. First, the results are very similar, and the main trends can be seen in both models. Second, there is an advantage in favour of the discriminative model (RefMethod), for the classification rates of the Known, Unknown, and Background categories (seen only for the harder Motorbikes dataset). This emphasises the importance of discriminability, when training the classifiers. In contrast, there is no difference in the classification of Unseen objects. This can be easily explained by the fact that these objects were not seen at training stage, thus there is no advantage for discriminative methods.

3.5 Discriminative Specific Classifiers Improve Performance

In this section I tested the importance of using a discriminative approach, by comparing our approach for building discriminative specific-level classifiers to less discriminative ones. In all variants of the models, the same general level classifier was used.

CurMethod Since this method uses only positive examples when learning the object models, these object models remain the same under all the variants. The negative examples are used only to learn the classification threshold, which distinguishes the scores of the known from unknown samples.

I have examined 3 variants for choosing the threshold of each object class model, as described in Section 2.1.1.

1. *Exploiting knowledge of sibling relations* - the most discriminative variant, where the threshold is trained to achieve the EER (Equal Error Rate) between the known class and its siblings. Classification rates for this variant are shown in Fig. 3.3.

[4] As already mentioned, the comparison methods are not accurate, as the models were not tuned to produce similar acceptance rates, and they differ in too many parameters. Nevertheless interesting points arise, which should be examined in future comparative research

2. *No knowledge of siblings relations* - a less discriminative variant, where the threshold is trained to achieve the EER between the known class and background images. The background images are the same set of negative samples used for training the general level classifier. Classification rates for this approach are shown in Fig. 3.4.

3. *Pure Generative* - the non discriminative variant using no negative examples, where the threshold is trained to achieve a predefined acceptance rate of the known class samples. Here the choice of a threshold is somewhat arbitrary, and different choices yield very different results for different classes of objects, depending on the real distribution of the known and unknown classes. Classification rates for this approach are shown in Fig. 3.4.

From the point of view of novelty detection, this last variant is the classical one-class classifier [49, 53, 38, 39], where during training only positive examples of a single class are given. Nevertheless, when we view the whole scheme as described in Algorithm 2.4 where the pure generative model is used in conjunction with the general level classifier, this is already a step beyond the traditional approach.

As can be seen in the comparison of Fig. 3.4 and Fig. 3.3, the results for the less-discriminative variants are usually inferior to the discriminative variants. Moreover these variants show a much less consistent behaviour, when different classes are left as unknown, and between different hierarchies. For example, in Fig. 3.4 in the case of the Faces dataset, we see good results with no knowledge of sibling relations, while the opposite is true with the exact same settings when applied to the Motorbikes data set.

This inconsistency proves what we expected, namely, that discriminative knowledge helps. In other words, in order to achieve an optimal threshold for separating one class from other classes, it is not enough to model the first class' distribution. The performance is highly related to the other classes' distribution as well. Clearly, for example, you cannot use the same threshold when separating motorbikes from fruits, as when separating sport-motorbikes from road-motorbikes. In the first case a high threshold (say acceptance ratio of 99%) will probably yield good results, while in the second one a more conservative threshold must be used (otherwise all the road-motorbikes will be classified as sport-motorbikes).

An important question arises here: even though the discriminative power proved to be helpful for separating similar classes, will it be advantageous in every level of the

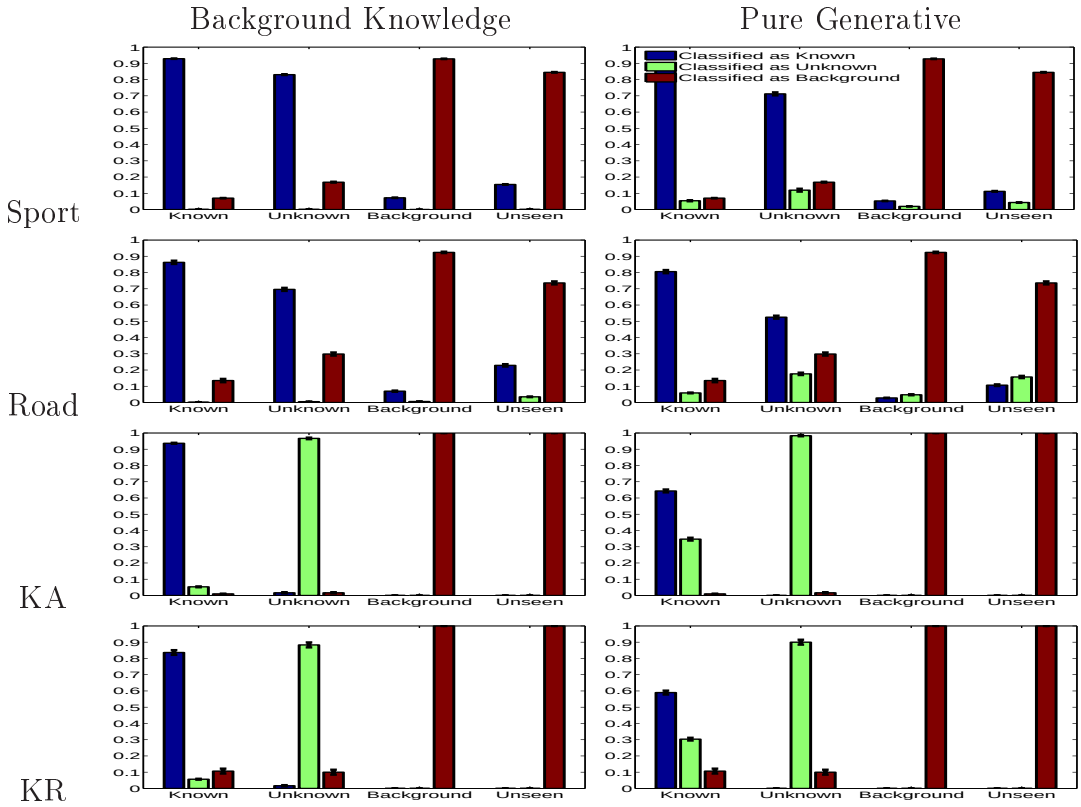


Figure 3.4: The non-discriminative variants of CurMethod. The left column shows results with no knowledge of sibling relations (only of background classes), and the right column shows the pure generative approach as explained in Section 3.5. We show representative results of the classes that were left out as unknown. From top to bottom: Cross (Motorbikes), Road (Motorbikes), KA (Faces), and KR (Faces). The meaning of the bars in every plot is the same as in Fig. 3.3.

hierarchy? Will it outperform the generative approach even in the more abstract levels, when separating classes that do not share similar shape or appearance, such as vehicles from fruits and animals? This is an open question that should be addressed in future researches.

RefMethod For comparison I present the results that were achieved for similar experiments with RefMethod. This method relies on both positive and negative examples during the training phase (see Section 2.1). The amount of discriminative information used for building the specific level classifiers, was varied by choosing different sets of

examples as the negative training set: 1) *1vsSiblings - Exploiting knowledge of sibling relations*, the most discriminative variant, where all train samples of the known sibling classes are used as the negative set when training each specific known class classifier. 2) *1vsBck - No knowledge of siblings relations*, a less discriminative variant, where the negative set of examples is similar to the one used when training the general level classifier, thus no information about the hierarchy is used in the model.

Applying these different variants when training with RefMethod, results in entirely different object models. Thus the known vs. unknown classification results depict different ROC curves, as shown in Fig. 3.5. Specifically, when comparing the *1vsSiblings* to *1vsBck* curves in Fig. 3.5, it can be seen that for all choices of classes, left out as the unknown, the corresponding ROC curve of the *1vsSiblings* method shows much better discrimination of known from unknown classes. This demonstrates that discriminative training with the sibling classes as negative samples significantly enhances performance.

3.6 Novel Class Detector is Specific

To test the validity of our novel class detection algorithm, we need to verify that it does not mistakenly detect low quality images, or totally unrelated novel classes, as novel sub-classes. Thus, two types of miss-classifications were examined, as presented in the following paragraphs.

Unseen Objects First, the algorithm was tested on samples of objects from classes, which are not related to the general class and had not been shown during the training phase. These samples are denoted *Unseen*, for which any classification other than rejection by the general level classifier is false. Thus, we expect most of these objects to be rejected. Moreover we expect the rate of false classification of unrelated classes as unknown classes to be similar to the rate of false classification as known classes.

The first prediction depends only on the quality of separation of the general level classifiers. The latter prediction is reasonable when the specific level classifiers are trained in a discriminative fashion to distinguish between siblings classes, and not between the known class and everything else (e.g. background or unrelated classes).

As Fig. 3.3 shows, by far most unseen samples are correctly rejected by the general level classifier. On the other hand, in both datasets (both with CurMethod and

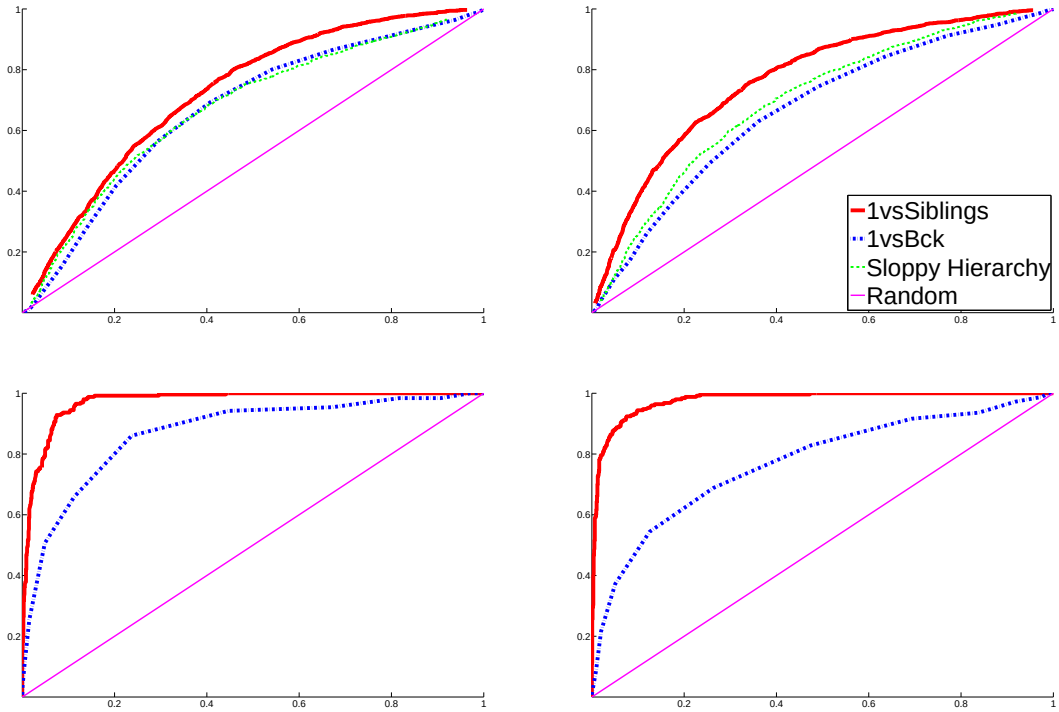


Figure 3.5: ROC curves showing True-Unknown classification rate on the Y-axis vs. False-Unknown Classification rate on the X-axis. The plot is based only on examples that were accepted by the General level classifier. Results are shown for RefMethod. *1vsSiblings* denotes the most discriminative training protocol, where specific class object models are learned using the known siblings as the negative set. *1vsBck* denotes the less discriminative training protocol, where the set of negative samples is the same as in the training of the General level classifier. Sloppy-Hierarchy denotes the case where the hierarchy was built using the procedure described in Section 3.7. Each plot shows results for a different class left out as the unknown, from left to right and top to bottom respectively: 'Cross-Motorbikes', 'Sport-Motorbikes', 'KA' Face and 'KL' Face. We only show two representative cases for each dataset, as the remaining cases look very similar.

RefMethod) there is a tendency to prefer the unknown classification in the cases of misclassification. In the Faces dataset this is less evident, but there might be too few misclassification cases to derive a clear conclusion. In the Motorbikes dataset this tendency is clear, and even more evident in CurMethod compared to RefMethod. These results are consistent with the assumption that RefMethod is more discriminative than CurMethod in the specific classifiers level, but more experiments have to be carried out

in order to deduct confident conclusions.

Noisy Images Second, to test the recognition of low quality images, I took images of objects from known classes and added increasing amounts of Gaussian white noise to the images. As can be seen in Figure 3.3, almost all the background images are rejected correctly by the general level classifier, and the addition of noise maintains this level of rejection. On the other hand the fraction of known objects classified correctly, decreases as the noise level increases.

In Figure 3.6 I examine the changes in the misclassification of samples from the known class as the level of noise increases. The goal is to see whether a larger fraction is misclassified as unknown class or as background.

Specifically, the ratio (FU-FB)/(FU+FB) is shown, where FU denotes false classification as unknown class, and FB denotes false classification as background. The higher this ratio is, the higher the ratio of unknown class misclassification to background misclassification is. An increase in the false identification of images as unknown should correspond to higher values of this ratio.

We would like this ratio to remain fixed, or at least not to grow, what will indicate that lower quality images do not tend to be classified as unknown (novel) classes. In fact, in Figure 3.6 the opposite trend can be seen: this expressions decreases with noise. Thus, at least as it concerns low quality images due to Gaussian noise, our model does not identify these images as coming from novel classes.

Interestingly, when running the same experiments from [63] with CurMethod, the levels of noise used for RefMethod were not informative enough. By this I mean that even with 15% addition of Gaussian noise, there were still very few wrong classifications of known class images. Nevertheless increasing the levels of noise some more (up to 25%) yielded very similar effects also with CurMethod, as can be seen in Figure 3.6. The high variance on the Faces results in CurMethod is due to very low number of wrong classifications, and a lower number of test repetitions compared to RefMethod.

3.7 Sloppy Hierarchical Relation

In order to explore the significance of hierarchy in our proposed scheme, I followed the same procedure, described in section 2, using a "sloppy" hierarchy. This way I can

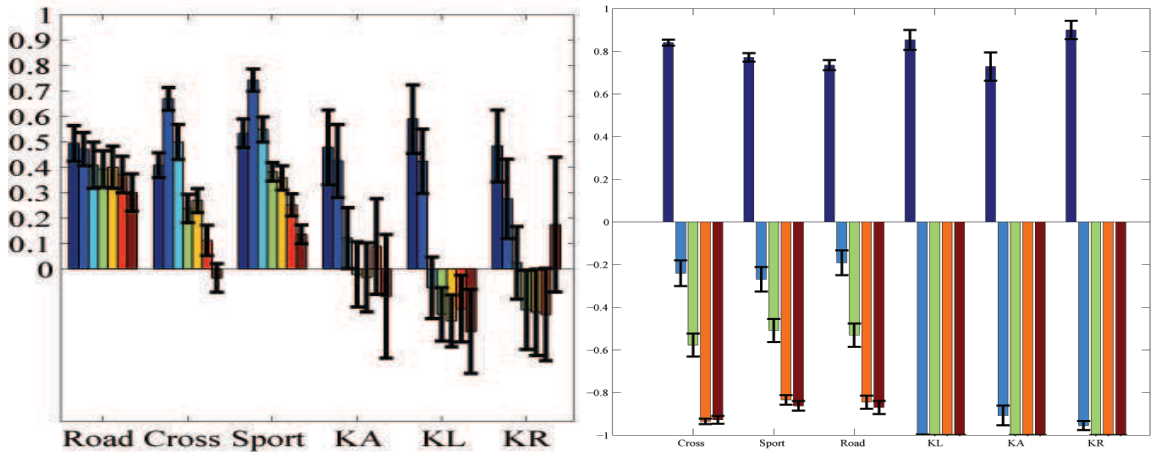


Figure 3.6: This figure shows the effect of adding Gaussian noise on the false classification rate of samples from known classes. Each bar shows the average over all experiments of (FU-FB)/(FU+FB), where FU denotes the false classification of known objects as unknown, and FB denotes the false classification of known objects as background. Lower values correspond to higher tendency to classify known class samples as background rather than as unknown class. Results are shown for CurMethod on the left, and RefMethod on the right. Each group of bars holds the results for a different class, left out as the unknown. In each group of bars, the different color-bars correspond to increasing levels of noise, from the left-most bar with no noise to the right-most bar with the highest amount of noise (25% for CurMethod and 15% for RefMethod).

compare the results with the "strict" hierarchy to results with the "sloppier" hierarchy. Only one thing was changed: instead of using a group of sub-classes that are strictly hierarchically related, a random group of sub-classes was collected. All the other steps remained unchanged. For instance, instead of using 'Sport Motorbikes' and 'Road Motorbikes' as the known group, when 'Cross Motorbikes' is the unknown class, we may choose 'Owl' and 'Cactus' as the known group with the same 'Cross Motorbikes' as the unknown class (see 3.7 for a visualization of possible object class hierarchies).

The acceptance rate by the general level classifier using the strictly built hierarchy compared to the sloppy hierarchy is shown in Figure 3.8. Results are shown for samples belonging to the known classes, unknown-class and background images.

As can be seen in Figure 3.8, the general level classifier which is learned for the sloppy hierarchy is less strict, in the sense that more background images are falsely accepted and more known and unknown images are falsely rejected by the general level classifier.

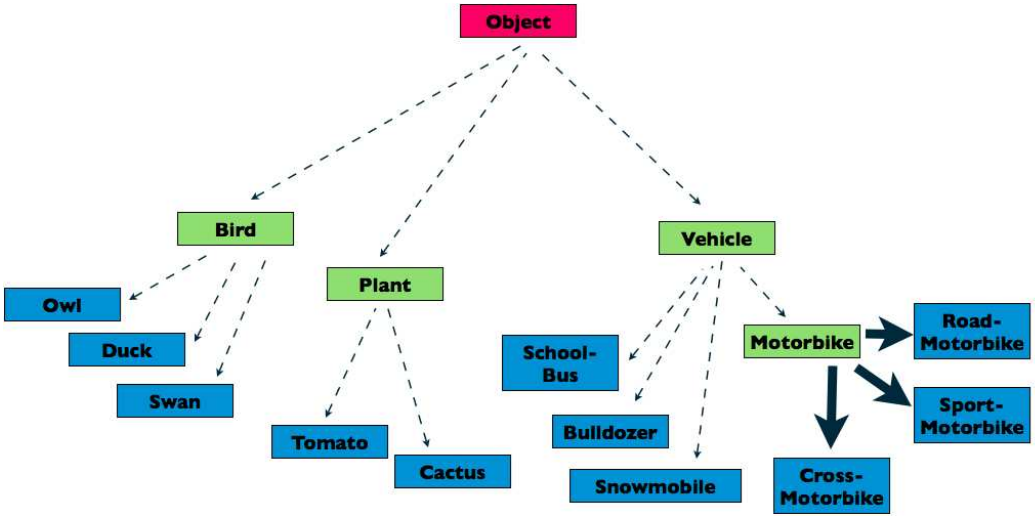
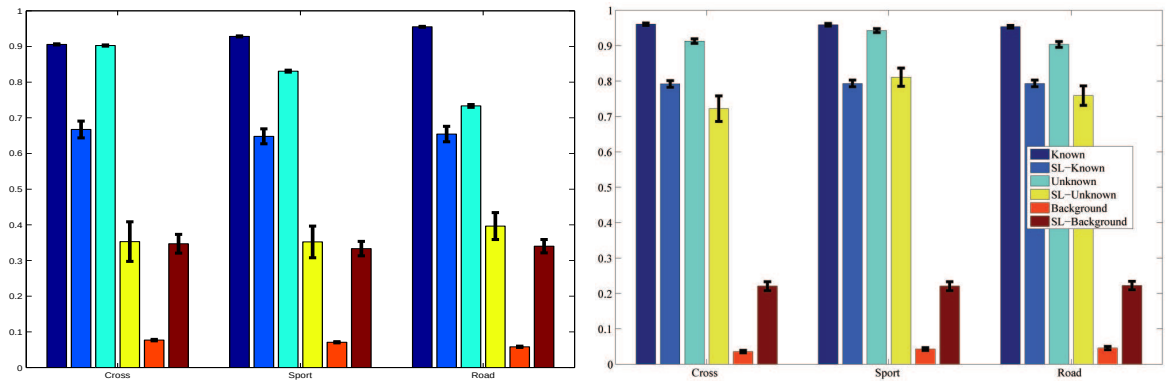


Figure 3.7: An illustration of strict vs. sloppy hierarchical relations to the different motorbikes in the hierarchy. Strictly related classes are classes which are connected by a parent only via full bold lines. Classes which are connected to the motorbike sub-classes via a dotted line are considered a sloppy hierarchical relation.

Figure 3.5 displays the ROC curves of the correct unknown classification versus false unknown classification, for samples that were accepted by the general level classifier. This was done for both strict and sloppy hierarchies using RefMethod. Here we can see that the distinction between known classes and an unknown class is significantly improved with the strict hierarchy as compared to the sloppy hierarchy.

These results indicate that a better general level classifier can be trained, when given access to a strict hierarchy, compared to some sloppier hierarchy. Thus, combining a better general level classifier with the specific level classifier, as described in Algorithm 2.4, clearly improves the identification of unknown classes by the full system.



CurMethod RefMethod

Figure 3.8: Acceptance rates of the General level classifier, with Strict and Sloppy hierarchies. Rates are presented for three categories: Known classes, Unknown class, and Background images. Two classifiers are compared, the first using a Strict Hierarchy and the second using a Sloppier Hierarchy. Six bars show, from left to right respectively: Strict Hierarchy - Known classes ('Known'), Sloppy Hierarchy - Known classes ('SL-Known'), Strict Hierarchy - Unknown class ('Unknown'), Sloppy Hierarchy - Unknown class ('SL-Unknown'), Strict Hierarchy - Background ('Background'), Sloppy Hierarchy - background ('SL-Background'). Results are shown for the cases where the Cross-Motorbikes, Sport-Motorbikes or Road-Motorbikes are left as the unknown class, from left to right respectively. The left column displays results for CurMethod, while the right column shows the output of RefMethod

Chapter 4

Summary

4.1 Overview

In this work I addressed the problem of novel object class recognition: how can we tell that a new input image contains an instance of a class we have never seen before? Previous approaches usually identify a novel class simply by having it rejected by **all** known classifiers, or by using prior information about the distribution of the novel classes.

Unfortunately, the second approach cannot be taken in many real-world scenarios, due to lack of knowledge about the distribution of the novel classes. While the first approach can be always used, it has two major drawbacks: it might be computationally expensive, as the number of known classes can grow to hundreds, thousands, and even more; it lacks the ability to distinguish between a novel interesting object class and clutter images, or uninteresting objects.

In the presented approach we exploit the hierarchical organization of objects and in particular existing hierarchical relations among the known classes, and propose a hierarchical discriminative algorithm, which detects novelty based on the disagreement between two levels of classifiers: some general level classifier accepts the new image, while all specific classifiers, belonging to the same sub-hierarchy, reject it.

I have analysed the properties of the proposed algorithm, showing the following results:

1. Basic efficiency of the algorithm was shown using a new implementation, which was tested on two different datasets.

2. Adding discriminative quality to the specific level classifiers, by exploiting the hierarchy, improves the performance.

3. The algorithm correctly detects relevant novel classes, and is not easily fooled by low quality images, or unrelated objects.

4. Using a strict hierarchy improves the results of the detector.

5. The generality of the new approach was demonstrated by implementing two very different object models as the baseline modules of the algorithm. As expected, the current results were similar to the previous implementation by Zweig [63].

4.2 Generative vs. Discriminative Implementation

As already mentioned throughout this work, comparing the two different methods is very problematic, nevertheless we can learn some interesting aspects of our approach from this comparison, and get some useful future thoughts. It was shown that the two object models use different training samples, different basic features, a different number of features, and different classification techniques. All these features influence the performance, and contribute to the variations of the results between the two models. We shall list some of the important features seen in the results:

1. The basic results (see section 3.4) show an advantage to RefMethod over CurMethod in almost all categories. This is true even when the general level classifiers show comparable performance. The reason for that can be the fact that RefMethod takes advantage of the hierarchy in a better way, by incorporating the discriminative features of the specific level classes in the object models. CurMethod, on the other hand, uses this knowledge only when setting the classification threshold. A support to this claim is the fact that no such advantage was seen for the Unseen category.

2. When testing the influences of different levels of "discriminativity" (see 3.5), we can not fairly compare RefMethod and CurMethod, since they have different outputs. RefMethod introduces different models, thus displays different ROC curves. RefMethod has a single model and differs only in the classification threshold -

different points on the same ROC curve. I will address this problem, and a few possible solutions in the last section.

3. In the noise results (see 3.6) we can see similar tendencies, although CurMethod is much less sensitive to noise. This is probably thanks to the usage of many simple low level features in a statistical fashion in CurMethod, while RefMethod uses a few very distinctive features (thus more sensitive to modifications, such as noise).

As we said, this comparison is far from being sufficient, but it can give us some food for thoughts about future research.

4.3 Future Research

This work presents a new topic in the research of hierarchical object recognition in general, and specifically of novelty detection. This topic is the investigation of the advantages and limitations of discriminative and generative properties. The first task is to build a framework, in which we have control over the discriminative and generative levels. In other words, a single implementation, where we can set different levels of discriminative power, while keeping everything else fixed.

A few possible solutions, using the CurMethod implementation are: using discriminative codebooks as the basis of the generative model; or using a discriminative separator over the generative representation of the model.

Once we have such a framework we can confront some new questions: Is the discriminative property useful at any level in the classes hierarchy, or maybe only at certain levels? How can we tell which is the best type of classifier for a certain level in the hierarchy? How strict should the hierarchy be in order to gain top performance? Obviously not too large and not too small, but how should it be chosen? How should we build the hierarchy that will produce the best classification results? All these and more are all interesting topics for future research in this field.

Bibliography

[1] M. Aly, P. Welinder, M. Munich, and P. Perona. Automatic Discovery of Image Families: Global Vs. Local Features. In *International Conference on Image Processing (ICIP), Cairo, Egypt, November 2009*, 2009.

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

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

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

[5] P. N. Belhumeur, J. P. Hespanha, and D. J. Kriegman. Eigenfaces vs. fisherfaces: Recognition using class specific linear projection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(7):711–720, August 1997.

[6] S. Belongie and J. Malik. Matching with shape contexts. In *CBAIVL '00: Proceedings of the IEEE Workshop on Content-based Access of Image and Video Libraries (CBAIVL'00)*, page 20. IEEE Computer Society, 2000.

[7] C. M. Bishop and J. Lasserre. Generative or discriminative? getting the best of both worlds. *BAYESIAN STATISTICS*, 8:3–24, 2007.

[8] G. Borgefors. Hierarchical chamfer matching: A parametric edge matching algorithm. *IEEE Trans. Pattern Anal. Mach. Intell.*, 10(6):849–865, 1988.

[9] A. Bosch, A. Zisserman, and X. Muoz. Scene classification using a hybrid generative/discriminative approach. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 30(4):712–727, April 2008.

[10] G. Bouchard and B. Triggs. The trade-off between generative and discriminative classifiers. In *IASC International Symposium on Computational Statistics (COMPSTAT)*, pages 721–728, August 2004.

[11] O. Chapelle, P. Haffner, and V. N. Vapnik. Support vector machines for histogram-based image classification. *Neural Networks, IEEE Transactions on*, 10(5):1055–1064, 1999.

[12] S. Chopra, R. Hadsell, and Y. LeCun. Learning a similarity metric discriminatively, with application to face verification. *Proc. of Computer Vision and Pattern Recognition Conference*, 2005.

[13] T. F. Cootes, G. J. Edwards, and C. J. Taylor. Comparing active shape models with active appearance models. In *BMVC*, 1999.

[14] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *Computer Vision and Pattern Recognition (CVPR)*, volume 1, pages 886–893, 2005.

[15] C. P. Diehl and J. B. Hampshire-Ii. Real-time object classification and novelty detection for collaborative video surveillance. In *In Proceedings of the International Joint Conference on Neural Networks*, pages 2620–2625, 2002.

[16] L. Fei-Fei, R. Fergus, and P. Perona. Learning generative visual models from few training examples: An incremental bayesian approach tested on 101 object categories. In *IEEE CVPR Workshop of Generative Model Based Vision (WGMBV)*, 2004.

[17] P. Felzenszwalb and D. Huttenlocher. Pictorial structures for object recognition. *Intl. Journal of Computer Vision*, 61(1):55–79, January 2005.

[18] R. Fergus, P. Perona, and A. Zisserman. Object class recognition by unsupervised scale-invariant learning. In *In CVPR*, pages 264–271, 2003.

[19] R. Fergus, P. Perona, and A. Zisserman. Weakly supervised scale-invariant learning of models for visual recognition. *International Journal of Computer Vision (IJCV)*, 71(3):273–303, 2007.

[20] M. Fink. Object classification from a single example utilizing class relevance metrics. In *NIPS*, 2004.

[21] M. Fischler and R. Elschlager. The representation and matching of pictorial structures. C-22:67–92, 1973.

[22] M. Fritz, B. Leibe, B. Caputo, and B. Schiele. Integrating representative and discriminant models for object category detection. In *In ICCV*, pages 1363–1370, 2005.

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

[24] T. T. Herbert Bay, Andreas Ess and L. V. Gool. Surf: Speeded up robust features. In *Computer Vision – ECCV*, pages 404–417. 2006.

[25] A. Holub, M. Welling, and P. Perona. Hybrid generative-discriminative object recognition. In *International Journal of Computer Vision (IJCV)*, volume 77, pages 239–258, 2007.

[26] F. J. Huang and Y. LeCun. Large-scale Learning with SVM and Convolutional for Generic Object Categorization. In *CVPR '06: Proceedings of the 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 284–291. IEEE Computer Society, 2006.

[27] T. Jebara. *Discriminative, generative and imitative learning.* PhD thesis, 2002. Supervisor-Pentland, Alex P.

[28] T. Kadir and M. Brady. Saliency, scale and image description. In *International Journal of Computer Vision (IJCV)*, volume 45, pages 83–105, 2001.

[29] Y. Ke and R. Sukthankar. Pca-sift: A more distinctive representation for local image descriptors. *Computer Vision and Pattern Recognition, IEEE Computer Society Conference on*, 2:506–513, 2004.

[30] A. Lanitis, C. J. Taylor, and T. F. Cootes. A unified approach to coding and interpreting face images. In *ICCV*, 1995.

[31] J. Lasserre. *Hybrid of Generative and Discriminative Methods for Machine Learning.* PhD thesis, 2008. Supervisor-Bishop, Christopher M.

[32] B. Leibe, A. Leonardis, and B. Schiele. Combined Object Categorization and Segmentation With An Implicit Shape Model. In *In ECCV workshop on statistical learning in computer vision*, pages 17–32, 2004.

[33] B. Leibe, A. Leonardis, and B. Schiele. Robust object detection with interleaved categorization and segmentation. *International Journal of Computer Vision (IJCV)*, 77(1):259–289, 2008.

[34] K. Levi, M. Fink, and Y. Weiss. Learning from a small number of training examples by exploiting object categories. In *Computer Vision and Pattern Recognition Workshop*, volume 6, page 96, Washington, DC, USA, 2004. IEEE Computer Society.

[35] R. S. Liu Yang, Rong Jin and F. Jurie. Unifying discriminative visual codebook generation with classifier training for object category recognition. In *CVPR*, 2008.

[36] D. G. Lowe. Object recognition from local scale-invariant features. *Computer Vision, IEEE International Conference on*, 2:1150–1157 vol.2, August 1999.

[37] M. Lyons, S. Akamatsu, M. Kamachi, and J. Gyoba. Coding facial expressions with gabor wavelets. *ICAFGR*, 1998.

[38] M. Markou and S. Singh. Novelty detection: a review-part 1: statistical approaches. *Signal Processing*, 83(12):2481–2497, 2003.

[39] M. Markou and S. Singh. Novelty detection: a review-part 2: neural network based approaches. *Signal Processing*, 83(12):2499–2521, 2003.

[40] D. Marr and H. K. Nishihara. Representation and recognition of the spatial organization of three-dimensional shapes. *Proceedings of the Royal Society of London:Biological Sciences*, 200(1140):269–294, 1978.

[41] M. Marszałek and C. Schmid. Semantic hierarchies for visual object recognition. In *Conference on Computer Vision & Pattern Recognition*, 2007.

[42] J. Matas, O. Chum, M. Urban, and T. Pajdla. Robust wide baseline stereo from maximally stable extremal regions. In *In British Machine Vision Conference*, volume 1, pages 384–393, 2002.

[43] J. Matas, M. Hamouz, K. Jonsson, J. Kittler, Y. Li, C. Kotropoulos, A. Tefas, I. Pitas, T. Tan, H. Yan, et al. Comparison of face verification results on the XM2VTS database. *ICPR*, 2000.

[44] K. Mikolajczyk and C. Schmid. Scale & affine invariant interest point detectors. *International Journal of Computer Vision (IJCV)*, 60(1):63–86, 2004.

[45] F. Moosmann, B. Triggs, and F. Jurie. Fast discriminative visual codebooks using randomized clustering forests. In *Neural Information Processing Systems (NIPS)*, 2006.

[46] P. Moreels and P. Perona. Evaluation of features detectors and descriptors based on 3d objects. *International Journal of Computer Vision (IJCV)*, 73(3):263–284, 2007.

[47] A. Y. Ng and M. I. Jordan. On discriminative vs. generative classifiers: A comparison of logistic regression and naive bayes. pages 841–848, 2001.

[48] M. Osadchy and E. Morash. Loose shape model for discriminative learning of object categories. 2008. IEEE Conf. on Computer Vision and Pattern Recognition.

[49] B. Scholkopf, R. Williamson, A. Smola, J. Shawe-Taylor, and J. Platt. Support vector method for novelty detection. *NIPS*, 2000.

[50] T. Serre, L. Wolf, S. Bileschi, and M. Riesenhuber. Robust object recognition with cortex-like mechanisms. *IEEE Trans. Pattern Anal. Mach. Intell.*, 29(3):411–426, 2007.

[51] J. Sivic, B. C. Russell, A. Zisserman, W. T. Freeman, and A. A. Efros. Unsupervised discovery of visual object class hierarchies. In *CVPR*, 2008.

[52] E. B. Sudderth, A. B. Torralba, W. T. Freeman, and A. S. Willsky. Learning hierarchical models of scenes, objects, and parts. In *ICCV*, pages 1331–1338, 2005.

[53] D. Tax and R. Duin. Support Vector Data Description. *Machine Learning*, 54(1):45–66, 2004.

[54] T. Tuytelaars and K. Mikolajczyk. *Local Invariant Feature Detectors: A Survey*. Now Publishers Inc., Hanover, MA, USA, 2008.

[55] I. Ulusoy and C. Bishop. Comparison of generative and discriminative techniques for object detection and classification. In *CLOR*, pages 173–195, 2006.

[56] V. Viitaniemi and J. Laaksonen. Techniques for image classification, object detection and object segmentation. In *VISUAL*, pages 231–234, 2008.

[57] P. A. Viola and M. J. Jones. Robust real-time face detection. *International Journal of Computer Vision*, 57(2):137–154, 2004.

[58] L. Wang. Toward a discriminative codebook: Codeword selection across multi-resolution. In *CVPR*, 2007.

[59] D. Weinshall, H. Hermansky, A. Zweig, J. Luo, H. Jimison, F. O, and M. Pavel. Beyond Novelty Detection: Incongruent Events, when General and Specific Classifiers Disagree. *NIPS*, 2008.

[60] J. Winn, A. Criminisi, and T. Minka. Object categorization by learned universal visual dictionary. In *ICCV '05: Proceedings of the Tenth IEEE International Conference on Computer Vision*, pages 1800–1807. IEEE Computer Society, 2005.

[61] D. Yeung and C. Chow. Parzen-window network intrusion detectors. *ICPR*, 2002.

[62] Z. Zhang, S. Chan, and L.-T. Chia. Codebook+: A new module for creating discriminative codebooks. In *ICME*, pages 815–818, 2007.

[63] A. Zweig and D. Weinshall. Exploiting Object Hierarchy: Combining Models from Different Category Levels. *ICCV*, 2007.