Learning from Few Examples for Visual Recognition Problems

Erik Rodner

Dissertation

zur Erlangung des akademischen Grades
doctor rerum naturalium (Dr. rer. nat.)

vorgelegt dem Rat der Fakultät für Mathematik und Informatik
der Friedrich-Schiller-Universität Jena

Reviewers:

1. Prof. Dr.-Ing. Joachim Denzler, Friedrich-Schiller-Universität Jena
2. Prof. Dr. rer. nat Christoph Schnörr, Universität Heidelberg
3. Prof. Josef Kittler, Ph.D., University of Surrey, United Kingdom

Submission of the thesis: 25th August 2011
Day of the oral defense: 2nd December 2011

Acknowledgments

First of all, I would like to thank Prof. Joachim Denzler for his support and trust during my years as a PhD student. This work would not have been possible without his guidance. Furthermore, Dr. Herbert Süße was and still is my mathematical oracle and it is always a pleasure to discover with him the theoretical connections between current and early computer vision algorithms.

I also owe a debt of gratitude to Michael Kemmler for so many fruitful discussions and Esther Wacker for the interesting cooperation in the wire rope area. In addition, I also thank my former diploma students Alexander Lütz, Björn Fröhlich, and Paul Bodesheim, who have broaden my scientific horizon with their topics and who are now colleagues with whom I really enjoy working with. In general, everyone of the computer vision research group in Jena contributed to a great scientific and friendly atmosphere, which might be one of the most important aspects necessary for completing a PhD. I am additionally indebted to Prof. Josef Kittler and Prof. Christoph Schnörr for reviewing my thesis.

Finally, I would like to thank my wife Wiebke and my family for their love, their constant encouragement, and for taking care of my work life balance, which was especially in danger the days directly before a conference deadline.

Abstract

The lack of training data is one of the core problems and limiting factors in many industrial and high-level vision tasks. Despite the human ability of quickly generalizing from often just a single example of an object category, current computer vision systems require a large number of visual examples to learn from. The following work deals with this problem and two paradigms are considered to tackle it: *transfer learning* and *one-class classification*.

The idea of *transfer learning* is that prior information from related tasks can be used to improve learning of a new task with few training examples. In this work, several transfer learning approaches are presented, which concentrate on transferring different types of knowledge from related tasks. The presented regularized tree method extends decision tree methods by incorporating prior knowledge from previously built trees of other tasks. Another proposed algorithm transfers feature relevance information to guide the search for suitable base classifiers during learning of a random decision forest. In addition, a third developed approach utilizes dependent Gaussian processes and allows for non-parametric transfer learning. Furthermore, a technique is presented that automatically selects a related task from a large set of tasks and estimates the required degree of transfer. The method is able to adaptively learn in heterogeneous environments and is based on efficient leave-one-out estimates and semantic similarities . All methods are analyzed in different scenarios (binary and multi-class transfer learning) and are applied to image categorization. Significant performance benefits are shown in comparison with current transfer learning methods and to classifiers not exploiting knowledge transfer.

Another very difficult problem occurs when training data is only available for a single class. To solve this problem, new efficient *one-class classification* methods are presented, which are directly derived from the Gaussian process framework and allow flexible learning with kernels. The suitability of the proposed algorithms for a wide range of applications is demonstrated for image categorization and action detection tasks, as well as the demanding application of defect localization in wire ropes. The experimental results show that the proposed methods are able to outperform other one-class classification methods. In addition, the influence of kernel hyperparameters is investigated.

A further study analyzes the performance gain achieved by using multiple sensors (color camera and time-of-flight camera) for generic object recognition. The presented combined system leads to a superior recognition performance compared to previous approaches.

Zusammenfassung

Das maschinelle Lernen aus wenigen Beispielen ist ein wichtiges und entscheidendes Problem bei vielen visuellen Erkennungsaufgaben, besonders in industriellen Anwendungen. Dennoch benötigen aktuelle Verfahren, im Gegensatz zum Menschen, meistens Hunderte von Beispielbildern. Die vorliegende Arbeit beschäftigt sich mit diesem Problem und versucht dieses durch die Verwendung zweier Konzepte zu lösen: *Lerntransfer* und *Ein-Klassen-Klassifikation*.

Als *Lerntransfer* wird die automatische Verwendung von Vorwissen ähnlicher Aufgabenstellungen für das Erlernen einer neuen Aufgabe bezeichnet. In dieser Arbeit werden mehrere Verfahren vorgestellt, die versuchen dieses Konzept des maschinellen Lernens umzusetzen. Eine in dieser Arbeit vorgestellte Methode erweitert Entscheidungsbaumklassifikatoren um die Möglichkeit, Vorwissen von bereits erlernten Entscheidungsbäumen anderer Aufgaben zu verwenden. Ein weiterer Ansatz transferiert Informationen über die Merkmalsrelevanz . Des weiteren wird ein drittes Verfahren vorgestellt, welches auf Gaußprozessen basiert und daher einen nicht-parametrischen Wissenstransfer ermöglicht. Ein besonderer Vorteil dieser Methode ist es, Klassifikationsaufgaben von denen Wissen transferiert werden soll, automatisch auszuwählen und den Einfluss des Vorwissens zu adaptieren. Dies wird durch eine effiziente Modellselektion und der Verwendung von semantischen Ähnlichkeiten ermöglicht. Alle Methoden werden im Rahmen der Bildkategorisierung ausgewertet. Die Ergebnisse zeigen einen signifikante Steigerung der Erkennungsleistung, im Vergleich zu aktuellen Methoden des Lerntransfers und Verfahren, welche keine zusätzlichen Lerndaten anderer Klassifikationsaufgaben verwenden.

Eine weitere wichtige Art von Aufgaben mit wenigen Lernbeispielen sind solche, bei denen nur Lerndaten für eine einzige Klasse vorhanden sind. Zur Lösung dieser Probleme werden neue Ansätze der *Ein-Klassen-Klassifikation* vorgestellt, welche direkt vom Konzept der Klassifikation mit Gaußprozessen abgeleitet werden. Die Nützlichkeit der Verfahren wird anhand der Bildkategorisierung, der Aktionserkennung und der schwierigen Aufgabenstellung der Defektlokalisierung bei Drahtseilen demonstriert. Die Ergebnisse der Experimente zeigen deutlich, dass die vorgestellten Methoden in der Lage sind, bessere Erkennungsergebnisse als bisherige Standardverfahren zu erzielen.

Ein zusätzlicher Aspekt dieser Arbeit ist die Entwicklung eines Systems zur generischen Objekterkennung, welches die Sensorinformationen einer Farb- und einer Time-of-Flight-Kamera kombiniert.

Contents

Chapter 1

Introduction

The following introductory chapter aims at motivating the thesis topic (Section 1.1) and explaining the basic principles used, such as knowledge transfer (Section 1.2) and one-class classification (Section 1.5). A large part concentrates on previous work related to this thesis, which is analyzed in a structured literature review (Section 1.3). The main contributions are summarized in Section 1.7 including an outline of the work.

1.1 Motivation

As humans we are able to visually recognize and name a large variety of object categories. A rough estimation of Biederman (1987) suggests that we know approximately 30.000 different visual categories, which corresponds to learning five categories per day, on average, in our childhood. Moreover, we are able to learn the appearance of a new category using few visual examples (Parikh and Zitnick, 2010). Despite the impressive success of current machine vision systems (Everingham et al., 2010), the performance is still far from being comparable to human generalization abilities. Current machine learning methods, especially when applied to visual recognition problems, often need several hundreds or thousands of training examples to build an appropriate *classification model* or *classifier*. This thesis tries to reduce this still existing gap between human and machine vision in visual learning scenarios.

The importance of efficient learning with few examples can be illustrated

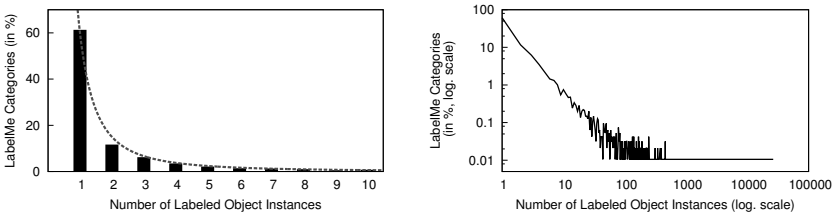


Figure 1.1: (a) Number of object categories in the LabelMe dataset for a specific number of labeled instances *(inspired by Wang et al. (2010))*; (b) Extended plot in logarithmic scale illustrating Zipf's law in the first part of the plot (Zipf, 1949). Similar statistics can be derived for *ImageNet* (Deng et al., 2009) and *TinyImages* (Torralba et al., 2008). The LabelMe database was obtained in April 2008.

by analyzing current large-scale datasets for object recognition, such as *LabelMe* (Russell et al., 2008; Torralba et al., 2010). Figure 1.1(a) shows the relative number of object categories in LabelMe that possess a specific number of labeled instances. A large percentage (over 60%) of all categories only have one single labeled instance. Therefore, even in datasets which include an enormous number of images and annotations in total, the lack of training data is a more common problem than one might expect. The plot also shows that the number of object categories with k labeled instances follows an exponential function $\beta \cdot k^{-\alpha}$, which is additionally illustrated in the right log-log plot of Figure 1.1. This phenomenon is known as *Zipf's law* (Zipf, 1949) and can be found in language statistics and other scientific data.

With current state-of-the-art approaches we are able to build robust object detectors for tasks with a large set of available training images, like detecting pedestrians or cars (Felzenszwalb et al., 2008). However, if we want to extract a richer semantic representation of an image, such as trying to predict different visual attributes of a car (model type, specific identity, etc.), we are likely not able to rely on a sufficient number of images for each new category. Therefore, high-level visual recognition approaches frequently suffer from weak training representations.

1.1.1 Industrial Applications

Problems related to a lack of learning examples are not restricted to visual object recognition tasks in real-world scenes, but are also prevalent in many industrial applications. Collecting more training data is often expensive, time-consuming or practically impossible. In the following, we give three examples tasks in which such a problem arises.

One important example is face identification (Tan et al., 2006), where the goal is to estimate the identity of a person from a given image. For example, such a system has to be trained with images of each person being allowed to access a protected security area. Obtaining hundreds of training images for each person is thus impractical, especially because the appearance should vary and include different illumination settings and clothing, which leads to a time-consuming procedure. Similar observations hold for writer or speaker verification (Yamada et al., 2010) and speech or handwritten text recognition (Bastien et al., 2010).

Another interesting application scenario is the prediction of user preferences in shop systems. The goal is to estimate the probability that a client likes a new product, given some previous product selections and ratings. If a machine learning system quickly generalizes from a few given user ratings and achieves a high performance in suggesting good products to buy, it is more probable that the client will use this shop frequently. In this application area, solving the problem of learning with few training examples is simply a question of cost. The economical importance of the problem can be seen in the *Netflix prize* (Bennett and Lanning, 2007), which promised one million dollars for a new algorithm which improves the rating accuracy of a DVD rental system. This competition has lead to a large amount of machine learning research related to *collaborative filtering*, which is a special case of *knowledge transfer* and is explained in more detail in Section 1.2.

A prominent and widely established field of application for machine learning and computer vision is *automatic visual inspection (AVI)* (Chin and Harlow, 1982). To achieve a high quality of an industrial production, several work pieces have to be checked for errors or defects. Due to the required speed and the high cost of manual quality control, the need for automatic visual defect localization arises. Whereas images from non-defective data can be easily obtained in large numbers, training images for all kinds of defects are often impossible to collect. A solution to solve this problem is to handle it as an *outlier detection* or *one-class classification* problem. In this case, learning data only consists of

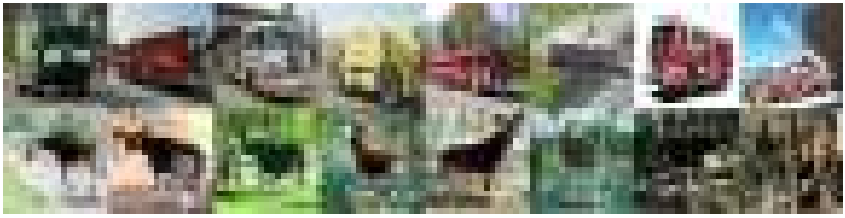


Figure 1.2: Images of two object categories (*fire truck* and *okapi*) from the *Caltech 256* database (Griffin et al., 2007)

non-defective examples and is used afterward to detect examples not belonging to the underlying distribution of the data. Previous work and related issues of this research topic are summarized in Section 1.5. An application is defect localization in wire ropes (Wacker and Denzler, 2011), which is also be studied in this thesis (Section 5.7).

1.1.2 Challenges

What are the challenges and the problems of traditional machine learning methods in scenarios with few training examples? First of all, we have to clarify our notion of "few". Common to all traditional machine learning methods are their underlying assumptions, which were formulated by Niemann (1990). The first postulate states:

> **Postulate 1:** *"In order to gather information about a task domain, a representative sample of patterns is available." (Niemann, 1990, Section 1.3, p. 9)*

Therefore, a scenario with few training examples can be defined as a classification task that violates this assumption by having an insufficient or non-representative sample of patterns. Of course, this notion depends on the specific application and on the complexity of the task under consideration.

One of the difficulties is the high variability in the data of high-level visual learning tasks. Some images from an object category database are given in Figure 1.2. A classification system has to cope with background clutter, different viewpoints, illumination changes and in general with a large diversity of the category (intra-class variance). On the one hand, this can only be performed with

a large amount of flexibility in the underlying model of the category, such as using a large set of features extracted from the images and a complex classification model. On the other hand, learning those models requires a large number of (representative) training examples. These conditions turn learning with few examples into a severe problem especially for high-level recognition tasks.

The trade-off between a highly flexible model and the number of training examples required can be explained quite intuitively for polynomial regression: Consider a set of n sample points of a one-dimensional m-order polynomial. The order of the polynomial is a measure of the complexity of the function. For the noise-free case, we need $n \geq m + 1$ examples to get an exact and unique solution. In contrast, noisy input data requires a higher number of examples to estimate a good approximation of the underlying polynomial. This direct dependency to the model complexity becomes more severe if we increase the input dimension D. The number of coefficients that have to be estimated, and analogous the number of examples required, grows polynomial in D according to $\binom{D+m}{m} = \mathcal{O}(D^m)$ (Bishop, 2006, Exercise 1.16). This immense increase in the amount of required training data is known in a broader as the *curse of dimensionality*.

A deeper insight and an analysis for classification rather than regression tasks is offered by the theoretical bounds derived in statistical learning theory for the error of a classification model or hypothesis h (Cucker and Smale, 2002; Cristianini and Shawe-Taylor, 2000). Assume that a learner selects a hypothesis from a possibly infinite set of hypotheses $\mathbb{H}$ which achieves zero error on a given sampled training set of size n. The attribute "sampled" refers to the assumption that the training set is a sample from the underlying joint distribution of examples and labels of the task. Due to this premise the following bound is not valid for one-class classification. A theorem proved by Shawe-Taylor et al. (1993, Corrolary 3.4) states that with probability of $1 - \delta$ the following number of training examples is sufficient for achieving an error below ϵ:

$$n \geq \frac{1}{\epsilon(1 - \sqrt{\epsilon})} \left(2 \ln \left(\frac{6}{\epsilon} \right) \dim(\mathbb{H}) + \ln \left(\frac{2}{\delta} \right) \right) \quad . \tag{1.1}$$

The term $\dim(\mathbb{H})$ denotes the *VC dimension*[1] of $\mathbb{H}$ and can be regarded as a complexity measure of the set of available models or hypotheses. For example, the class of all D-dimensional linear classifiers including a bias term has a VC

[1] VC is an abbreviation for Vapnik–Chervonenkis.

dimension of $D + 1$ (Vapnik, 2000, Section 4.11, Example 2). Let us now take a closer look on the bound in Eq. (1.1). If we fix the maximum error ϵ and choose an appropriate small value for δ, we can see that the sufficient number of training examples depends linearly on the VC dimension $\dim(\mathbb{H})$. This directly corresponds to our previous example of polynomial regression, because the VC dimension of D-variate polynomials of up to order m is exactly $\binom{D+m}{m}$ (Ben-David and Lindenbaum, 1998).

1.1.3 The Importance of Prior Knowledge

Nearly all machine learning algorithms can be formulated as optimization problems, whether in a direct way, such as done by *support vector machines (SVM)* (Cristianini and Shawe-Taylor, 2000), or in an indirect manner like *boosting* (Friedman et al., 2000) approaches. From this point of view, we can say that learning with few examples inherently tries to solve an ill-posed optimization problem. Therefore, it is not possible to find a suitable well-defined solution without incorporating additional (prior) information. The role of prior information is to (indirectly) reduce the set of possible hypotheses. For example, if we know in advance that for a classification task only the color of an object is important, *e.g.* if we want to detect expired meat, only a small number of features have to be computed and a lower dimensional linear classifier can be used. In this situation the VC dimension is reduced, which results in a lower bound for the sufficient number of training examples (*cf.* Eq. (1.1)).

Introducing common prior knowledge into the machine learning part of a visual recognition system is often done by regularization techniques that penalize non-smooth solutions (Schölkopf and Smola, 2001) or the "complexity" of the classification model. Examples of such techniques are L_2-regularization, also known as *Tikhonov regularization* (Vapnik, 2000), or L_1-regularization, which is mostly related to methods trying to find a sparse solution (Seeger, 2008).

Other possibilities to incorporate prior knowledge include *semi-supervised learning* and *transductive learning*, which utilize large sets of unlabeled data to support the learning process (Fergus et al., 2009). Unlabeled data can help to estimate the underlying data manifold and, therefore, are able to reduce the number of model parameters. However, the use of unlabeled data is not studied in this thesis.

1.2 Knowledge Transfer and Transfer Learning

Machine learning tasks related to computer vision always require a large amount of prior knowledge. For a specific task, we indirectly incorporate prior knowledge into the classification system by choosing image preprocessing steps, feature types, feature reduction techniques, or the classifier model. This choice is mostly based on prior knowledge manually obtained by a software developer from previous experiences on similar visual classification tasks. For instance, when developing an automatic license plate reader, ideas can be borrowed from optical text recognition or traffic sign detection. Increasing expert prior knowledge decreases the number of training examples needed by the classification system to perform a learning task with a sufficient error rate. However, this requires a large manual effort.

The goal of some techniques presented in this work is to perform transfer of prior knowledge from previously learned tasks to a new classification task in an automated manner, which is known as *transfer learning* and which is a special case of *knowledge transfer*. The advantage compared to traditional machine learning methods, or *independent learning*, is that we do not have to build new classification systems from the scratch or by large manual effort. Previously known tasks used to obtain prior knowledge are referred to as *support tasks* and a new classification task only equipped with few training examples is called *target task*. In the following, we concentrate on *inductive transfer learning* (Pan and Yang, 2010), which assumes that we have labeled data for the target and the support tasks. Especially interesting are situations where a large number of training examples for the support tasks exists and prior knowledge can be robustly estimated. Other terms for transfer learning are *learning to learn* (Thrun and Pratt, 1997), *lifelong learning* and *interclass transfer* (Fei-Fei et al., 2006).

The concept of transfer learning is also one of the main principles that explain the development of the human perception and recognition system (Brown and Kane, 1988). For example, it is much easier to learn Spanish if we are already able to understand French or Italian. The knowledge transfer concept is known in the language domain as language transfer or linguistic interference (Odlin, 1989). We already mentioned that a child quickly learns new visual object categories in an incremental manner without using many learning examples. Figure 1.3 shows some images of a transfer learning scenario for visual object recognition. Generalization from a single example of the new animal category, on the right hand side of Figure 1.3, is possible due to a large set of available

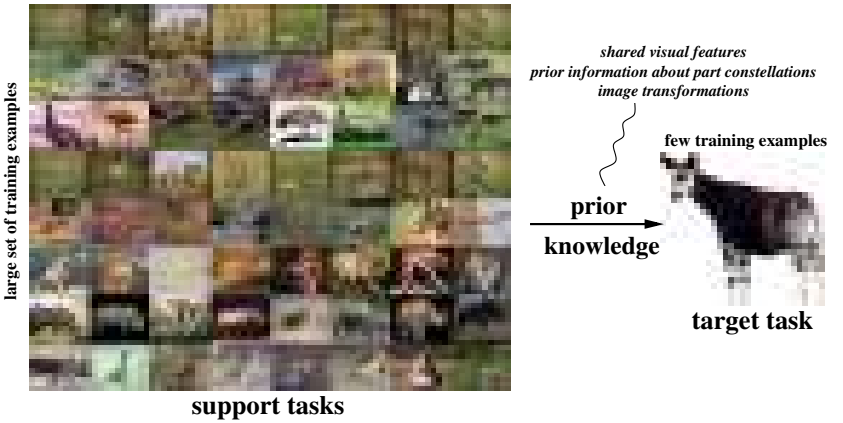


Figure 1.3: The basic idea of transfer learning for visual object recognition: a lot of visual categories share visual properties which can be exploited to learn a new object category from few training examples.

memories (images) of related animal categories. These categories often share visual properties, such as typical object part constellations (head, body and four legs) or similar fur texture.

Developing transfer learning techniques and ideas requires answering four different questions: *"What, how, from where and when to transfer?"*. First of all, the type of knowledge which will be transferred from support tasks to a new target task has to be defined, *e.g.* information about common suitable features. Detailed examples are listed in a paragraph of Section 1.3.1. The transfer technique applied to incorporate prior knowledge into the learning process of the target task strictly depends on this definition but is not determined by it. For example, the relevance of features for a classification task can be transferred using generalized linear models (Lee et al., 2007) or random decision forests (Section 3.3). Prior knowledge is only helpful for a target task if the support tasks are somehow related or similar. In some applications not all available previous tasks can be used as support tasks, because they would violate this assumption. Giving an answer to the question *"From where to transfer?"* means that the learning algorithm has to select suitable support tasks from a large set of tasks. These learning situations are referred to as learning in *heterogeneous*

environments. Of course, we expect that additional information incorporated by transfer learning always improves the recognition performance on a new target task, because it is the working hypothesis of transfer learning in general. However, in machine learning there is no guarantee at all that the model learned from a training set is also appropriate for all unseen examples of the task, *i.e.* there are no deterministic warranties concerning the generalization ability of a learner [2]. Therefore, knowledge transfer can fail and lead to worse performance compared to independent learning. This event is known as *negative transfer* and happens in everyday life. For example, if we use "false friends" when learning a new language, *e.g.* German speakers are sometimes confused about "getting a gift" because the word "Gift" is the German word for poison, which is likely not a thing you are happy to get. Situations in which negative transfer might occur are difficult to detect. We show in Section 3.4 how to handle this issue by estimating the degree of transfer knowledge needed.

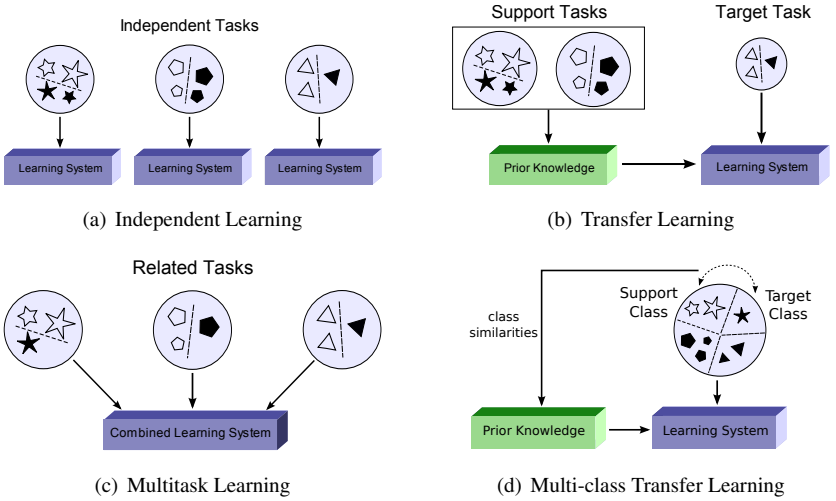
Besides transfer learning, another type of knowledge transfer is *multitask learning* which learns different classification tasks jointly[3]. Combined estimation of model parameters can be very helpful, especially if a set of tasks are given, with each having only a small number of training examples. In contrast to transfer learning there is no prior knowledge obtained in advance, but the model parameters of each task are coupled together. For example, given a set of classification tasks, relevant features can be estimated jointly and all classifiers are learned independently with the reduced set of features. A possible application is collaborative filtering as mentioned in Section 1.1. In the following thesis, we stick to transfer learning but borrow some ideas from multitask learning approaches.

Figure 1.4 summarizes the conceptual difference between independent learning, transfer learning, and multitask learning. Furthermore, we illustrate the principle of *multi-class transfer learning*. In contrast to all other approaches, transfer within a single multi-class task is considered rather than between several (binary) classification tasks. To emphasize this fact, we use the term target class rather than target task. The main difficulty is that the target class has to be distinguished from the support class, even though information was transferred and exploits their similarity.

Formal definitions are given in Section 3.1. It should be noted that another

[2] Note that even the bound in Eq. (1.1) only holds with probability $1 - \delta$.

[3] Xue et al. (2007) use the term symmetric multitask learning to refer to jointly learning tasks and asymmetric multitask learning refers to our use of the term transfer learning.



(a) Independent Learning

(b) Transfer Learning

(c) Multitask Learning

(d) Multi-class Transfer Learning

Figure 1.4: The concept of (a) traditional machine learning with independent tasks, (b) transfer learning, (c) multitask learning and (d) multi-class transfer learning *(inspired by Pan and Yang (2010))*.

area of knowledge transfer is *transductive transfer learning*, which concentrates on transferring knowledge from one application domain to another. For example, the goal is to recognize objects in low quality webcam images with the support of labeled data from photos made by digital cameras. Related terms are *sample-selection bias*, *covariate shift*, and *domain adaptation* (Pan and Yang, 2010). Although all of the ideas presented in this work could be applied to those scenarios, our applications concentrate on transfer learning in the same domain (Section 3.1) and the combination of multiple data from different domains or sensors to solve a single task (Section 5.8).

1.3 Previous Work on Transfer Learning

There is a large body of literature trying to handle the problem of learning with few examples. A lot of work concentrates on new feature extraction methods, or classifiers, which show superior performance to traditional methods espe-

cially for few training examples (Levi and Weiss, 2004). The few examples problem was also tackled by introducing manual prior knowledge of the application domain, such as augmenting the training data by applying artificial transformations (Duda et al., 2000; Bayoudh et al., 2007) also known as *data manufacturing*.

In the following, we concentrate on methods related to the principle of transfer learning and multitask learning as introduced in the previous section. Other similar surveys and literature reviews can be found in the work of Fei-Fei (2006) from a computer vision perspective and the journal paper of Pan and Yang (2010), which gives a comprehensive overview of the work done in the machine learning community. There is also a textbook by Brazdil et al. (2009) covering the broader area of *meta-learning*, and the edited book of Thrun and Pratt (1997), about the early developments of learning to learn.

1.3.1 What and how to transfer?

The type of knowledge which is transferred from support tasks to a target task is often directly coupled with the method used to incorporate this additional information. Therefore, we give a combined literature review on answers to both questions.

Learning Transformations One of the most intuitive types of knowledge which can be transferred between categories is application-specific transformations or distortions. While in data manufacturing methods these transformations have to be defined using manual supervision, transfer learning methods learn this information from support tasks.

For example, a face recognition application can significantly benefit from transformations transferred from other persons using optical flow techniques (Beymer and Poggio, 1995). Estimating latent transformations and distortions of images (e.g. illumination changes, rotations, translations) within a category is proposed by Miller et al. (2000) and Learned-Miller (2006) using an optimization approach. Their approach called *Congealing* tries to minimize the joint entropy of gray-value histograms in each pixel with a greedy strategy. The obtained transformations can be directly applied to the images of a target task and used in a nearest neighbor approach for text recognition. Restricting and regularizing the complexity of the class of transformations during estimation is important for a good generalization, because it additionally introduces generic

prior knowledge. The original Congealing approach proposed a heuristic normalization step directly applied during optimization. An extension without explicit normalization, and a study of different complexity measures, can be found in the work of Vedaldi et al. (2008) and Vedaldi and Soatto (2007). Huang et al. (2007) use Congealing to align images of cars or faces with local features.

Shared Kernel or Similarity Measure How we compare objects and images strictly depends on the current task. A distance metric or a more general similarity measure can be an important part of a classification system, e.g. in nearest neighbor approaches. The term *kernel* is a related concept which also measures the similarity between input patterns. Hence, a distance metric or a kernel function is an important piece of prior knowledge which can be transferred to new tasks.

The early work of Thrun (1996) used neural network techniques to estimate a similarity measure for a specific task. In general, the idea of estimating an appropriate metric from data is a research topic on its own called *metric learning* (Yang and Jin, 2006). A common idea is to find a metric which minimizes distances between examples of a single category (intra-class) and maximizes distances between different categories (inter-class). Applying a similarity measure to another task is straightforward when using a nearest neighbor classifier. Fink (2004) used the metric learning algorithm of Shalev-Shwartz et al. (2004), which allows online learning and estimates the correlation matrix of a Mahalanobis distance.

Metric learning for visual identification tasks is presented by Ferencz et al. (2008). They show how to find discriminative local features which can be used to compare different instances of an object category, *e.g.* distinguishing between specific instances of a car. In this work, metric learning is based on learning a binary classifier which estimates the probability that both images belong to the same object instance. The obtained similarity measure can be used for visual identification with only one single training example for each instance. Another application of metric learning is domain adaptation as explained in Section 1.2. The paper of Saenko et al. (2010) presents a new database for testing domain adaptation methods and also gives results of a metric learning algorithm. In contrast, Jain and Learned-Miller (2011) propose to perform domain adaptation by applying Gaussian process regression on scores of examples near the decision boundary.

Shared Features Visual appearance can be described in terms of features such as color, shape and local parts. Thus, it is natural to transfer information about the relevance of features for a given task. This idea can be generalized to shared base classifiers which allow modeling feature relevance. Fink et al. (2006) study combining perceptron-like classifiers of the support and target task. Due to the decomposition into multiple base classifiers (weak learners), ensemble methods and especially Boosting approaches (Freund and Schapire, 1997) are suitable for this type of knowledge transfer. Levi et al. (2004) extend the standard Boosting framework by integrating task-level error terms. Weak learners which achieve a small error on all tasks should be preferred to very specific ones. A similar concept is used in the work of Torralba et al. (2007), who propose learning multiple binary classifiers jointly with a Boosting extension called *JointBoost*. The algorithm tries to find weak learners shared by multiple categories. This also leads to a reduction of the computation time needed to localize an object with a sliding-window approach. Experiments of Torralba et al. (2007) show that the number of feature evaluations grows logarithmic in the number of categories, which is an important benefit compared to independent learning. An extension of this approach to kernel learning can be found in Jiang et al. (2007). Salakhutdinov et al. (2011) exploits category hierarchies and performs feature sharing by combining hyperplane parameters.

Shared Latent Space Finding discriminative and meaningful features is a very difficult task. Therefore, the assumption of shared features between tasks is often not valid for empirically selected features used in the application. Quattoni et al. (2007) propose assuming an underlying latent feature space which is common to all tasks. They use the method of Ando and Zhang (2005) to estimate a feature transformation from support tasks derived from caption texts of news images. To estimate relevant features, the subsequent work (Quattoni et al., 2008) propose an eigenvalue analysis and the use of unlabeled data.

Latent feature spaces can be modeled in a Bayesian framework using Gaussian processes, which leads to the so called *Gaussian process latent variable model (GP-LVM)* (Lawrence, 2005). Incorporating the idea of a shared latent space into the GP-LVM framework allows using various kinds of noise models and kernels (Urtasun et al., 2008).

Constellation Model and Hierarchical Bayesian Learning An approach for knowledge transfer between visual object categories was presented by Fei-Fei

et al. (2006). Their method is inspired by the fact that a lot of categories share common object parts which are often also in the same relative position. Based on a generative constellation model (Fergus et al., 2003) they propose using maximum a posteriori estimation to obtain model parameters for a new target category. The prior distribution of the parameters corresponding to relative part positions and part appearance is learned from support tasks. The underlying idea can also be applied to a shape based approach (Stark et al., 2009).

A prior on parameters shared between tasks is an instance of hierarchical Bayesian learning. Raina et al. (2006) used this concept to transfer covariance estimates of parts of the feature set.

Joint Regularization A lot of machine learning algorithms such as SVM are not directly formulated in a probabilistic manner but as optimization problems. These problems often include regularization terms connected to the complexity of the parameter, which would correspond to a prior distribution in a Bayesian setting. The equivalent to hierarchical Bayesian learning as described in the last paragraph is a joint regularization term shared between tasks.

Amit et al. (2007) propose using *trace norm regularization* of the weight matrix in a multi-class SVM approach. They show that this regularization is related to the assumption of a shared feature transformation and task-specific weight vectors with independent regularization terms. Instead of transferring knowledge between different classification tasks this work concentrates on transfer learning in a multi-class setting, *i.e.* multi-class transfer.

Sharing a low dimensional data representation for multitask learning is the idea of Argyriou et al. (2006). The proposed optimization problem learns a feature transformation and a weight vector jointly and additionally favors sparse solutions by utilizing an $L_{1,2}$ regularization. Multitask learning with kernel machines was first studied by Evgeniou et al. (2005). Their idea is to reduce the multitask problem to a single task setting by defining a combined kernel function or *multitask kernel* and a new regularizer.

Shared Prior on Latent Functions The framework of Gaussian processes allows modeling a prior distribution of an underlying latent function for each classifier (Rasmussen and Williams, 2005) using a kernel function.

If we want to learn a set of classifiers jointly in a multitask setting, an appropriate assumption is that all corresponding latent functions are sampled from the same prior distribution. Lawrence et al. (2004) suggest learning the

hyperparameters of the kernel function jointly by maximizing the marginal likelihood. This idea was also applied to image categorization tasks (Kapoor et al., 2010). A more powerful way of performing transfer learning is studied with multitask kernels originally introduced by Evgeniou et al. (2005). Bonilla et al. (2007) use a parameterized multitask kernel that is the product of a base kernel comparing input features and a task kernel modeling the similarity between tasks and using meta or *task-specific features*. Task similarities can also be learned without additional meta features by estimating a non-parametric version of the task kernel matrix (Bonilla et al., 2008). A theoretical study of the generalization bounds induced by this framework can be found in Chai et al. (2008). Schwaighofer et al. (2005) propose an algorithm and model to learn the fully non-parametric form of the multitask kernel in a hierarchical Bayesian framework.

The *semi-parametric latent factor model (SLFM)* of Teh et al. (2005) is directly related to a multitask kernel. The latent function for each task is modeled as a linear combination of a smaller set of underlying latent functions. Therefore, the full covariance matrix has a smaller rank, which directly corresponds to the rank assumption of other transfer learning ideas (Amit et al., 2007). A more general framework which allows modeling arbitrary dependencies between examples and tasks using a graph-theoretic notation is presented by Yu and Chu (2008).

Semantic Attributes and Similarities Transfer learning with very few examples of the target task can be difficult due to the lack of data to estimate task relations and similarities correctly. Especially if no training data (neither labeled nor unlabeled) is available, other data sources have to be used to perform transfer learning. This scenario is known as *zero-shot learning* and uses the concept of *learning with attributes*, an area which received much attention in recent years. The term attribute refers to category-specific features.

Lampert et al. (2009) use a large database of human-labeled abstract attributes of animal classes (e.g. brown, stripes, water, eats fish). One idea is to train several attribute classifiers and use their output as new meta features. This representation allows recognizing new categories without real training images only by comparison with the attribute description of the category. The knowledge transferred from support tasks is the powerful discriminative attribute representation which was learned with all training data. A similar idea is presented by Larochelle et al. (2008) for zero-shot learning based on task-specific features. A

theoretical investigation of zero-shot learning with an attribute representation is given by Palatucci et al. (2009) and concentrates on analysis with the concept of *probable approximate correctness (PAC)* (Vapnik, 2000).

Instead of relying on human-labeled attributes, internet sources can be used to mine attributes and relations. The papers of Rohrbach et al. (2010a,b) compare different kinds of linguistic sources, such as WordNet (Pedersen et al., 2004), Google search, Yahoo and Flickr. A large-scale evaluation of their approach can be found in Rohrbach et al. (2011). Attribute based recognition can help to generalize to new tasks or categories (Farhadi et al., 2009) which is otherwise difficult using a training set only equipped with ordinary category labels. Attributes can also help to boost the performance of object detection rather than image categorization as shown in (Farhadi et al., 2010). Their transfer learning approach heavily relies on model sharing of object parts between categories.

Lampert and Krömer (2010) use a generalization of maximum covariance analysis to find a shared latent subspace of different data modalities. This can also be applied to transfer learning with attributes by regarding the attribute representation as a second modality. Beyond zero-shot learning semantic similarities are also used to guide regularization (Wang et al., 2010).

Context Information Up to now, we only covered transfer learning in which knowledge is used from visually similar object categories or tasks. However, dissimilar categories can also provide useful information if they can be used to derive contextual information. For example it is likely to find a keyboard next to a computer monitor, which can be a valuable information for an object detector. Methods using contextual information always exploit dependencies between categories and tasks and are therefore a special case for knowledge transfer approaches.

Fink and Perona (2003) propose training a set of object detectors simultaneously with an extended version of the *AdaBoost* algorithm (Viola and Jones, 2004) and can be regarded as an instance of multitask learning. In each round of the boosting algorithm the map of detection scores is updated and used as an additional feature in subsequent rounds. A similar idea is presented by Shotton et al. (2008) for *semantic segmentation*, which labels each pixel of the image as one of the learned categories. The work of Hoiem et al. (2005) pursues the same line of research, but clearly separates the support and target tasks. In a first step geometric properties of image areas are estimated. The resulting labeling into planar, non-planar, and porous objects, as well as ground and sky areas

can be used to further assist local detectors as high-level features. Contextual relationships between different categories can also be modeled directly with a *conditional Markov random field (CRF)* as done by Rabinovich et al. (2007) and Galleguillos et al. (2008) on a region-based level for semantic segmentation.

1.3.2 Heterogeneous Transfer: From Where to Transfer?

Automatically selecting appropriate support tasks from a large set is a difficult sub-task of transfer learning. Therefore, most of the previous work presented in this thesis so far assumes that support tasks are given in advance. An exception is the early work of Thrun and O'Sullivan (1996), which proposes the *task clustering* algorithm. Similarities between two tasks are estimated by testing the classifier learned on one task using data from the other task. Afterward, clustering can be performed with the resulting task similarity matrix.

Mierswa and Wurst (2005) select relevant features for a target task by comparing the weights of the SVM hyperplane with each of the available tasks. Therefore, the algorithm selects a similar but more robustly estimated feature representation. The work of Kaski and Peltonen (2007) performs transfer learning with logistic regression classifiers and models the likelihood of each task as a mixture of the target task likelihood and a likelihood term which is independent of all other tasks. Due to the task-dependent weight, the algorithm can adapt to heterogeneous environments. In general, selecting support tasks is a model selection problem, therefore, techniques like leave-one-out are used (Tommasi and Caputo, 2009; Tommasi et al., 2010). Heterogeneous tasks can also be handled within the regularization framework of Argyriou et al. (2006) by directly optimizing a clustering of the tasks (Argyriou et al., 2008).

1.4 Requirements of Visual Transfer Learning

We have seen that there are a large number of previous works on transfer learning covering very different aspects of the topic. However, we can extract some of the main properties and requirements of visual transfer learning:

1. A probabilistic formulation of the classification technique offers the possibility to directly incorporate prior knowledge of related tasks by modeling the prior distribution of classifier parameters.

2. Sharing and transferring information about relevant features is an intuitive way to perform transfer learning for visual object recognition tasks.

3. Non-parametric learning with kernels is especially beneficial for visual recognition tasks, because it allows handling categories with a large variability but a limited number of training examples.

4. The algorithm should automatically adjust the amount of transfer information or the strength of regularization.

5. Selecting support tasks automatically is important to allow for transfer learning in heterogeneous learning scenarios.

6. Multi-class transfer learning has not received much attention from the research community, although some early approaches (Miller et al., 2000; Fink, 2004) can be applied to certain application areas of this principle.

The algorithms presented in this thesis each concentrate on certain points of this list.

1.5 One-class Classification

If training examples for one category in a binary classification task are not available, classification corresponds to *one-class classification*. Other terms are *outlier detection*, *concept learning* or *novelty detection* (Bishop, 2006). Given a set of learning examples of a single category, which are often referred to as positive examples, the goal is to classify new examples as either belonging to the category known or to a completely new category. Instead of hard classification decisions, one-class classifiers often estimate a scoring function, which tells us something about the outlier likeliness of an unseen example. The related task of *density estimation* additionally requires the scoring function to be a well-defined probability density, normalizing to one with respect to all inputs, and being non-negative in the whole domain.

Application scenarios often arise due to the difficulty of obtaining training examples for rare cases, such as images of production failures in defect localization tasks (Hodge and Austin, 2004) or image data from non-healthy patients in medical applications (Tarassenko et al., 1995). In these cases, one-class classification offers to describe the distribution of positive examples and to treat

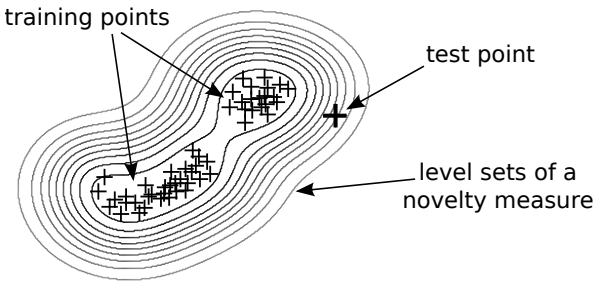


Figure 1.5: The principle of one-class classification and outlier detection: the distance of a test point to the training set is calculated with a novelty measure, which relates to the underlying data distribution.

negative examples as outliers, which can be detected without explicitly learning their appearance.

Aside from the lack of training data, other situations in which one-class classification is beneficial, are cases where the variety of one category is rather large. An example for typical application areas is object detection, which is often formulated as a binary classification task with an object category and a background category corresponding to "all other images". Obtaining a representative and unbiased set of examples of such a background category is difficult (Chen et al., 2001; Lai et al., 2002). One-class classification can also be important in a multi-class scenario to detect data from new categories. This is especially important in applications such as bacteria recognition (Dundar et al., 2009). Without handling outlier data, the decision of the multi-class classifier would be erroneous and could lead to a completely wrong analysis.

The different challenges explained in Section 1.1.2, *e.g.* model complexity vs. number of sufficient training examples and the curse of dimensionality, also arise in one-class classification domains and are unfortunately more severe (Tax, 2001). One of the reasons is that there is no way to approximate the expected error of the classifier on future data using empirical error estimates. Therefore, two trivial solutions are always possible and fit to the data provided: Either every unseen example is accepted as being a positive one or a new example is only accepted if it is equal to one of the training points. Without smoothness, normalization or volumetric constraints, one-class classification is an ill-posed problem, even with a large training set (Vapnik, 1995, Section 1.8.2, p. 26-27).

1.6 Previous Work on One-class Classification

A common way to perform one-class classification is to utilize a parameterized density function and maximize the model likelihood (or posterior), with respect to the set of density parameters. For example, a data distribution can be modeled as a normal distribution and the learning examples are used to estimate the mean vector and the covariance matrix (Bishop, 2006). An obvious generalization are *Gaussian mixture models (GMM)* (Bishop, 2006), which allow for modeling data distributions with more than one mode.

Despite their usage in a large number of applications, parametric methods always assume a special shape of the underlying distribution. Gaussian mixture models can offer some flexibility by increasing the number of mixture components, but this has to be paid with a high-dimensional parameter vector. An elegant solution is to use non-parametric methods, which use the available training dataset directly without an intermediate parameter estimation step. One idea, which is known as *kernel density estimation* or *Parzen estimator* (Scott and Sain, 2004), is to smooth the empirical data distribution with a window function or kernel. Tax and Duin (2004) present an approach called *support vector data description (SVDD)*, which estimates a minimal enclosing sphere around the data distribution. Modeling arbitrary shaped distributions can be done by transforming the input data into a high-dimensional feature space by using the *kernel trick*. A highly related algorithm is ν*-SVM* developed by Schölkopf et al. (2001), which finds a hyperplane separating the data from the origin of the feature space. Adams et al. (2009) propose a density estimation method based on Gaussian processes (Rasmussen and Williams, 2005). They have to rely on *Markov chain Monte Carlo (MCMC)* techniques to handle the involved normalization of the scoring function. The approach of Roth (2006) uses a *kernel fisher discriminant (KFD)* classifier to perform one-class classification. In contrast to the majority of other papers, the author also presents how to perform model selection. Hoffmann (2007) utilizes the reconstruction error of the kernel version of PCA to judge on the novelty of new examples.

Another idea to handle one-class classification tasks is to generate examples of the negative class in an artificial manner. Dundar et al. (2009) suggest sampling the parameters of normal distributions representing the negative class from a prior estimated using the models of several positive classes. This special setting is useful for detecting new categories of for instance Bacteria spectra (Akova et al., 2010).

1.7 Contribution and Outline of this Thesis

Chapter 2 presents fundamental machine learning concepts needed to formulate the new ideas and approaches developed in this thesis. We hence concentrate on classification techniques such as random decision forests and kernel machines like SVM and Gaussian process classifiers. The main part of this thesis is Chapter 3 and the presentation of experiments and applications in Chapter 5, which include the following contributions of this thesis:

- We present a transfer learning extension of random decision forests, which can be used for multi-class and binary transfer learning (Section 3.2).
- We show how to transfer information about relevant features from one task to another by developing a second modification of the learning algorithm of random decision forests (Section 3.3).
- A transfer learning algorithm utilizing Gaussian process models is presented, which is able to automatically select suitable support tasks and to adapt the influence of the transferred information (Section 3.4).
- We show how semantic similarity measures, such as the ones derived from the WordNet database (Pedersen et al., 2004), can be utilized to further improve learning of a task with few training examples (Section 3.4.4).
- Several new one-class classification (OCC) methods are derived from the Gaussian process framework allowing non-parametric novelty detection and easy computation (Section 3.5).
- We demonstrate the benefits of our transfer learning methods and compare them to previous approaches in a large set of image categorization experiments (Section 5.2 and Section 5.3).
- All of the proposed OCC approaches are evaluated in image categorization applications and compared to state-of-the-art methods like support vector data description of Tax (2001) (Section 5.5).
- We show how to apply our OCC methods to the task of defect localization in wire ropes (Section 5.7) and action detection (Section 5.6).
- Learning of visual object categories with few training examples significantly benefits from multiple sensors. We present a system for generic

object recognition that uses a time-of-flight and a standard CCD camera and which significantly outperforms previous approaches for this task (Section 5.8).

Chapter 4 describes computer vision related methods used in our experiments to calculate features and kernel functions. The thesis concludes in Chapter 6 with a summary, a discussion of the thesis results, as well as several ideas for future research directions.

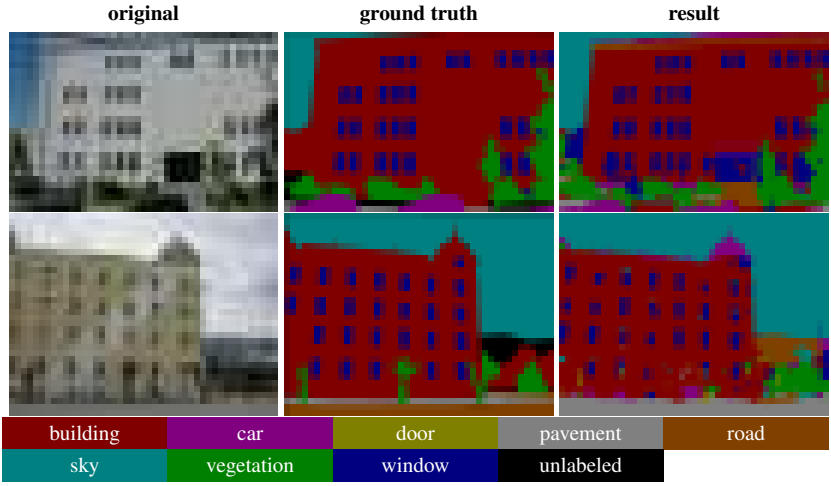
1.8 Related Contributions

A part of the research done during the time of my PhD studies is not included in this thesis and only a short summary is given in the following. Some of the work originated from diploma theses I have supervised.

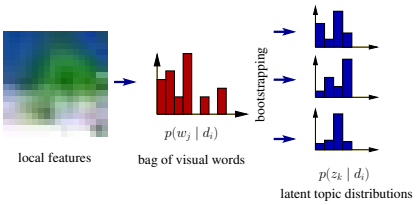
In Kemmler et al. (2009), we analyze the suitability of the combination of a time-of-flight and a standard CCD camera for scene recognition tasks similar to our studies of generic object recognition with the same hardware setup (Section 5.8). A large set of feature types and classifiers is evaluated and compared and we also study the necessity of preprocessing the range image preceding the feature calculation. Even though the task is heavily ill-posed and difficult in general, the resulting recognition accuracy is more than 65% for seven really challenging scene categories with similar looking office rooms.

Scene recognition can also be done by utilizing topic models, such as *probabilistic Latent Semantic Analysis (pLSA)* introduced by Hofmann (2001). In Rodner and Denzler (2009b), we propose to learn several topic models with a random fraction of the training data, which is similar to the Bagging idea of Breiman (1996) (Section 2.2.3). A visual outline of this idea is depicted in Figure 1.6(b). With our randomized technique, we are able to improve the classification accuracy of the scene recognition approach of Bosch et al. (2008), which utilized pLSA as a feature transformation technique. The experiments additionally show the inferior performance of topic model methods compared to plain bag-of-visual-words approaches in the case of few training examples.

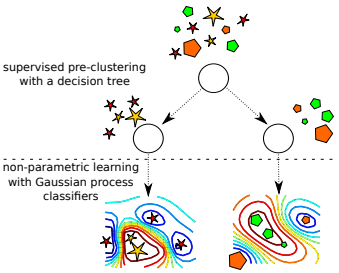
In Fröhlich et al. (2010), we demonstrate the ability of random decision forests and local color features to perform *facade recognition*, which is a semantic segmentation task and requires labeling each pixel as belonging to a specific category, *e.g.* door, building and window (Figure 1.6(a)). The interesting result of this work is that an additional time-consuming feature transformation technique,



(a) Results of our semantic segmentation approach (Fröhlich et al., 2010): (left) original image; (middle) ground truth segmentation; (right) result of our approach. All images are colored according to their labels. Therefore, the figure should be viewed in the online version.



(b) Outline of our randomized pLSA approach presented in Rodner and Denzler (2009b)



(c) Outline of our approach to large-scale Gaussian process classification (Fröhlich et al., 2010).

Figure 1.6: Some visual results and schematic outlines of the approaches developed during my PhD studies that are *not presented* in this thesis.

like the one proposed by Csurka and Perronnin (2008), is not necessary when using random decision forests.

Learning a Gaussian process classifier or even a regressor requires a cubical computation time with respect to the training examples. This fact prevents these techniques from being used in large-scale learning tasks, such as semantic segmentation applications. Therefore, we proposed in Fröhlich et al. (2011) to use a random decision forest as a pre-clustering method and learn a Gaussian process classifier only for the small set of training examples corresponding to a leaf node (Figure 1.6(c)).

Chapter 2

Machine Learning

The following chapter reviews fundamental concepts of state-of-the-art machine learning algorithms often applied in computer vision. We assume that the reader already has some insights in probability and information theory, otherwise the introductory sections of Bishop (2006) are a good starting point. An experienced reader familiar with decision trees, kernel machines and non-parametric Bayesian methods like Gaussian processes should take a quick look in the first section to understand the notations used in subsequent chapters of this thesis.

2.1 Mathematical Preliminaries and Notation

In the following, we define several terms related to machine learning and formulate the goal of traditional machine learning methods that *do not* exploit knowledge transfer. An extension to the mathematical framework including transfer learning algorithms is given in Section 3.1.

Machine learning is always about finding the latent relation between *observations* or *inputs* $\boldsymbol{x} \in \mathcal{X}$, *e.g.* images or D-dimensional feature vectors, and *outputs* $y \in \mathcal{Y}$, such as $\mathcal{Y} = \mathbb{R}$ for regression and $\mathcal{Y} = \{-1, 1\}$ for binary classification tasks. If we consider a classification task, $\mathcal{Y}$ is a finite set with elements referred to as *classes* or *categories* and we use the term *label* instead of output to refer to y. The unknown relationship between inputs and outputs can often be described by a function $\tilde{h} : \mathcal{X} \to \mathcal{Y}$. In case of a classification task, this function assigns a given example to one of the known categories or classes.

Given a *training set* $\mathcal{D} = \{(\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_n, y_n)\}$ of n training examples, the goal of the *learner* is to estimate the function $\tilde{h}$. Especially for binary classification tasks, instead of modeling $\tilde{h}$ directly, many approaches use a real-valued intermediate function $f : \mathcal{X} \to \mathbb{R}$ and apply a subsequent threshold operation, *e.g.* $h(\boldsymbol{x}) = \operatorname{sign}(f(\boldsymbol{x}))$. We use the notation $\operatorname{sign}(\cdot)$ as an abbreviation for the sign function, which is 1 for all positive values and -1 otherwise.

In view of a probabilistic perspective, a *learning task* is defined by an unknown joint distribution $p(y, \boldsymbol{x})$ of inputs and labels that determines their relation. For example, given a functional dependency, the joint distribution would be defined by $p(y, \boldsymbol{x}) = \delta\left[\tilde{h}(\boldsymbol{x}) - y\right] p(\boldsymbol{x})$ using the delta function $\delta[\cdot]$. The training set $\mathcal{D}$ is a sample of this distribution and we assume that each training example was sampled identically and independently from $p(y, \boldsymbol{x})$. The aim of the learner can be redefined in more general probabilistic terms as estimating the posterior $p(y_* \mid \boldsymbol{x}_*, \mathcal{D})$ of a yet unseen example $\boldsymbol{x}_* \in \mathcal{X}$. However, most machine learning algorithms concentrate on maximizing this posterior with respect to y_* to estimate the most likely label. For classification tasks, this corresponds to the function $\hat{h}$ which minimizes the probability of misclassification:

$$\hat{h}(\boldsymbol{x}_*) = \operatorname*{argmin}_{k \in \mathcal{Y}} p(y \neq k \mid \boldsymbol{x} = \boldsymbol{x}_*, \mathcal{D}) \tag{2.1}$$

$$= \operatorname*{argmin}_{k \in \mathcal{Y}} \left(1 - p(y = k \mid \boldsymbol{x} = \boldsymbol{x}_*, \mathcal{D})\right) \tag{2.2}$$

$$= \operatorname*{argmax}_{k \in \mathcal{Y}} p(y = k \mid \boldsymbol{x} = \boldsymbol{x}_*, \mathcal{D}) \ . \tag{2.3}$$

In the remaining part of this thesis, we skip the specification of the random variable if the meaning is non-ambiguous, *e.g.* $p(\boldsymbol{x} = \boldsymbol{x}_*)$ is shortened to $p(\boldsymbol{x}_*)$ and we use $\boldsymbol{x}$ as a notation of a single input and the corresponding random variable.

If we know the exact posterior, we could make an optimal decision according to the label. Unfortunately, in most cases we do not know anything about the posterior, except the information available from the learning examples. Thus, we have to rely on modeling the posterior in an appropriate way. This is done directly by *discriminative* classifiers, such as support vector machines or Gaussian process classifiers. In contrast, *generative* classifiers estimate the posterior indirectly by modeling the conditional likelihood of inputs $\boldsymbol{x}$ and by applying

Bayes theorem:

$$p(y_* \mid \boldsymbol{x}_*, \mathcal{D}) = \frac{p(\boldsymbol{x}_* \mid y_*, \mathcal{D})\; p(y_* \mid \mathcal{D})}{p(\boldsymbol{x}_* \mid \mathcal{D})} \tag{2.4}$$

$$= \frac{p(\boldsymbol{x}_* \mid y_*, \mathcal{D})\; p(y_* \mid \mathcal{D})}{\sum_{y \in \mathcal{Y}} p(\boldsymbol{x}_* \mid y, \mathcal{D})\; p(y \mid \mathcal{D})} \quad . \tag{2.5}$$

The striking difference is that generative approaches try to estimate the distribution of the input data for each category instead of predicting the probability of a label. The denominator in Eq. (2.4) is irrelevant if only the maximum of the posterior is of interest.

On the one hand, generative approaches have the advantage that a model of the data distribution is available which can be used to detect outliers (Section 1.5) or to sample new data points. The second property is especially useful for model verification. On the other hand, discriminative methods directly solve the problem without using a bypass. This view is manifested in the following famous quotation of Vladimir Vapnik:

> *"When solving a given problem, try to avoid solving a more general problem." (Vapnik, 1995, Section 1.9, p.28)*

The approaches presented in this thesis are mostly based on discriminative approaches, hence, we prefer the discriminative view and skip analogous equations and explanations for generative models in the next more conceptual section.

2.2 Estimation Theory and Model Selection

We have seen how a machine learning task can be defined in terms of probability theory. Therefore, solving classification problems can always be viewed as statistical inference and we review some basic concepts in the following section.

Let us assume that we have a set of probabilistic models, which are parameterized with $\boldsymbol{\theta} \in \Theta$. For example, $\boldsymbol{\theta}$ could consist of the mean and the variance of a simple normal distribution $\boldsymbol{\theta} = (\mu, \sigma^2)$, or determines the normal vector of a separating hyperplane. We denote a model of the posterior or likelihood by conditioning with respect to the parameters $\boldsymbol{\theta}$, *e.g.* $p(y \mid \boldsymbol{x}, \boldsymbol{\theta})$ for discriminative models and $p(\boldsymbol{x} \mid y, \boldsymbol{\theta})$ for generative models. For discriminative models, we additionally make use of $h(\boldsymbol{x}; \boldsymbol{\theta})$ as a notation of a deterministic model.

2.2.1 Maximum Likelihood and Maximum A Posteriori Estimation

Many machine learning algorithms proceed in two steps. The first step is the optimization of the model parameters during learning using some criterion. Afterward, the model equipped with the estimated parameters is used to predict the output of future data. A simple objective is the likelihood of the training data given the model parameters:

$$\hat{\boldsymbol{\theta}}^{\mathrm{ML}} \stackrel{\mathrm{def}}{=} \operatorname*{argmax}_{\boldsymbol{\theta} \in \Theta} p(\mathcal{D} \mid \boldsymbol{\theta}) \ , \tag{2.6}$$

which is known as *maximum likelihood (ML)* estimation. The optimization provides a so called *point estimate* that corresponds to the mode of the likelihood distribution. This formulation does not incorporate any prior probability of the model itself. Each parameter vector in the model space Θ is equally likely. The consequence is that ML estimation can quickly lead to overfitting, because complex models adapt to the learning data even though the resulting model is a priori unlikely. One way to reduce those effects is to incorporate prior knowledge about $\boldsymbol{\theta}$ by weighting the model likelihood with a prior $p(\boldsymbol{\theta})$. An equivalent formulation is the optimization of the parameter posterior:

$$\hat{\boldsymbol{\theta}}^{\mathrm{MAP}} \stackrel{\mathrm{def}}{=} \operatorname*{argmax}_{\boldsymbol{\theta} \in \Theta} p(\boldsymbol{\theta} \mid \mathcal{D}) = \operatorname*{argmax}_{\boldsymbol{\theta} \in \Theta} p(\mathcal{D} \mid \boldsymbol{\theta}) \; p(\boldsymbol{\theta}) \ , \tag{2.7}$$

also known as *maximum a posteriori (MAP)* estimation. Figure 2.1 illustrates the two types of estimation. The likelihood in Figure 2.1(a) has a high peak at $\boldsymbol{\theta} = 2$ with a small width, probably being a result of overfitting. If we incorporate a prior distribution of $\boldsymbol{\theta}$, the mode of the resulting posterior is shifted towards $\boldsymbol{\theta} = 1$ and the estimate of MAP estimation differs from the original ML estimate. Equivalence between both estimators arises in the presence of a uniform prior distribution, which is only defined properly for parameter spaces Θ with finite measure. The difficulty concerning MAP estimation lies in the definition of a suitable prior. In many practical applications the prior is chosen to be a member of a parametric family. Parameters related to the prior are often termed *hyperparameters* and are selected manually or estimated from related tasks as shown in Section 3.2.

After optimizing the underlying parameter vector, the *plug-in principle* is applied, which uses the resulting discriminative models $p\left(y_* \mid \boldsymbol{x}_*, \hat{\boldsymbol{\theta}}^{\mathrm{MAP}}\right)$ or

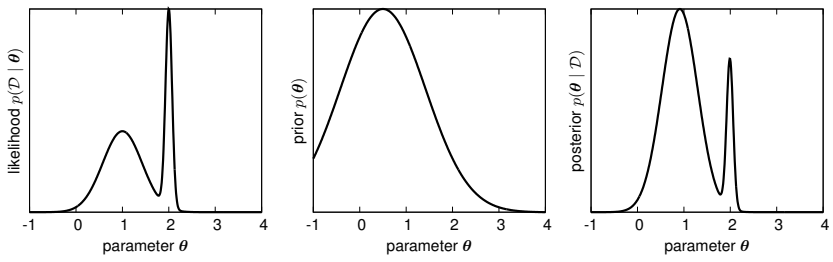


Figure 2.1: A simple example explaining maximum likelihood and maximum a posteriori estimation: (a) likelihood with a mode of small variance, (b) a normal distribution as a prior, (c) the resulting posterior with a different mode compared to the likelihood.

$p\left(y_* \mid \boldsymbol{x}_*, \hat{\boldsymbol{\theta}}^{\mathrm{ML}}\right)$ to predict labels of future data. The disadvantage of the plug-in principle is that the variance of the estimate is not taken into account.

2.2.2 Bayesian Inference through Marginalization

Exact Bayesian inference allows modeling the data likelihood or the posterior directly using the training data without intermediate parameter estimates, which is sometimes referred to as *non-parametric*[1]. Rather than using point estimates, exact Bayesian inference marginalizes the parameter $\boldsymbol{\theta}$ (the symbol $\overset{*}{=}$ highlights derivations which require certain assumptions):

$$p(y_* \mid \boldsymbol{x}_*, \mathcal{D}) = \int_{\Theta} p(y_*, \boldsymbol{\theta} \mid \boldsymbol{x}_*, \mathcal{D})\, d\boldsymbol{\theta} \tag{2.8}$$

$$\overset{*}{=} \int_{\Theta} p(y_* \mid \boldsymbol{x}_*, \boldsymbol{\theta})\, p(\boldsymbol{\theta} \mid \mathcal{D})\, d\boldsymbol{\theta} \tag{2.9}$$

$$= \int_{\Theta} p(y_* \mid \boldsymbol{x}_*, \boldsymbol{\theta}) \left(\frac{p(\mathcal{D} \mid \boldsymbol{\theta})\, p(\boldsymbol{\theta})}{p(\mathcal{D})}\right) d\boldsymbol{\theta}\ . \tag{2.10}$$

[1] Two definitions of the term "non-parametric" are common in the machine learning literature referring to situations in which (1) no parametric assumption is utilized or (2) the model complexity grows with the number of training examples. In this thesis, we use the first definition.

To derive the second equation, we assumed that the likelihood of a new example is conditionally independent of the training set $\mathcal{D}$ given the model parameters $\boldsymbol{\theta}$, which is also the main premise when using maximum likelihood or maximum a posteriori estimation. This assumption is quite natural, because the model should be developed such that every important information in the training set can be covered by model parameters. Minka and Picard (1997) refer to this assumption as $\boldsymbol{\theta}$ being a *separator*. An analogous formulation applies to the likelihood $p(\boldsymbol{x}_* \mid y_*, \mathcal{D})$. In contrast to the plug-in principle, exact Bayesian inference is much more general. This becomes obvious with a glance at the reduction of marginalization to the MAP or ML estimator in a very special case. If we consider a model posterior $p(\boldsymbol{\theta} \mid \mathcal{D})$ with a single impulse shifted to the mode $\hat{\boldsymbol{\theta}}^{\text{MAP}}$ of the original posterior, the integral can be evaluated directly:

$$p(y_* \mid \boldsymbol{x}_*, \mathcal{D}) = \int_{\Theta} p(y_* \mid \boldsymbol{x}_*, \boldsymbol{\theta}) \ \left(\delta \left[\boldsymbol{\theta} - \hat{\boldsymbol{\theta}}^{\text{MAP}}\right]\right) \ d\boldsymbol{\theta} \tag{2.11}$$

$$= p(y_* \mid \boldsymbol{x}_*, \hat{\boldsymbol{\theta}}^{\text{MAP}}) \ . \tag{2.12}$$

The delta function $\delta\,[\cdot]$ viewed as a density function has zero variance. Hence, point estimation of model parameters does not take the variance of the estimation into account. Figure 2.1(a) explains this very well by showing a likelihood with a high peak but a small width. In this case, the probability that the likelihood estimate represents the data is quite low contrary to the high likelihood value, which only represents a density value. Despite the fact that in our example in Figure 2.1 the incorporation of a prior distribution and the use of the MAP estimate prevents from taking a critical mode, it does not solve the problem in general. A slight shift of the prior leads to an equivalence of the ML and MAP estimate. In contrast, the resulting likelihood of the Bayesian approach in Eq. (2.9) can be viewed as an infinite sum of different model specific likelihoods each weighted by the model posterior. Thus, the Bayesian approach is robust to singularities like the one illustrated in Figure 2.1. The presented principle is also known as *Bayesian model averaging (BMA)*, although it sometimes refers to another perspective that concentrates on model selection and techniques trying to reduce the model space Θ (Hoeting et al., 1999).

Similar to the MAP estimator, the selection of a prior is a critical ingredient, not only because of the accuracy but also due to the computational tractability. The model parameter vector is often high-dimensional, which hinders the evaluation of the integral and requires techniques such as *Markov chain Monte*

Carlo (Bishop, 2006), *Laplace approximation* or *expectation propagation* (Rasmussen and Williams, 2005). Analytical solutions are only available in rare situations. Some parametric families of likelihood distributions have a corresponding parametric family of prior distributions that leads to a closed-form solution of the marginalization maintaining the parametric form of the likelihood. These priors are called *conjugate priors*.

Although we do not need to estimate a single suitable parameter to compute the posterior of an unknown label, sometimes it is appropriate to use the *Bayesian estimate* of the parameter. This estimate is defined to be the minimizer of the expectation of a loss $L : \Theta^2 \to \mathbb{R}$ with respect to the model posterior:

$$\hat{\boldsymbol{\theta}}^{\text{Bayes}} \stackrel{\text{def}}{=} \underset{\tilde{\boldsymbol{\theta}} \in \Theta}{\operatorname{argmin}} \; \mathbb{E}_{\boldsymbol{\theta}} \left(L(\boldsymbol{\theta}, \tilde{\boldsymbol{\theta}}) \,|\, \mathcal{D} \right) \tag{2.13}$$

$$= \underset{\tilde{\boldsymbol{\theta}} \in \Theta}{\operatorname{argmin}} \int_{\Theta} L(\boldsymbol{\theta}, \tilde{\boldsymbol{\theta}}) \cdot p(\boldsymbol{\theta} \,|\, \mathcal{D}) \; d\boldsymbol{\theta} \;\; . \tag{2.14}$$

For a quadratic loss, the Bayesian estimate is equivalent to the *minimum mean square error (MMSE)* or *minimum variance error* criterion (Denzler, 2003):

$$\hat{\boldsymbol{\theta}}^{\text{MMSE}} \stackrel{\text{def}}{=} \underset{\tilde{\boldsymbol{\theta}} \in \Theta}{\operatorname{argmin}} \; \mathbb{E}_{\boldsymbol{\theta}} \left(\left|\left| \boldsymbol{\theta} - \tilde{\boldsymbol{\theta}} \right|\right|^2 \,|\, \mathcal{D} \right) \;\; . \tag{2.15}$$

Differentiating with respect to $\tilde{\boldsymbol{\theta}}$ quickly leads to the fact that the resulting estimate is exactly the mean value of the model posterior:

$$\hat{\boldsymbol{\theta}}^{\text{MMSE}} = \mathbb{E}(\boldsymbol{\theta} \,|\, \mathcal{D}) = \int_{\Theta} \boldsymbol{\theta} \cdot p(\boldsymbol{\theta} \,|\, \mathcal{D}) \; d\boldsymbol{\theta} \;\; . \tag{2.16}$$

In case of a Gaussian model posterior, the MAP estimate is equivalent to the Bayesian estimate, which is obvious because the mean of a Gaussian is identical to its mode.

2.2.2.1 Comparison with Model Combination

At first glance, marginalization and the involved weighted averaging of different models looks like a generalization of weighted combinations of classifiers like those estimated by Boosting (Friedman et al., 2000). However, Bayesian inference through marginalization does not have model combination abilities (Minka,

2002). If the number of training examples increases, the entropy of the model posterior $p(\boldsymbol{\theta}|\mathcal{D})$ decreases and all weight is put on one single model even though a uniform weight would give a better accuracy. A detailed explanation of this phenomenon can be found in the work of Minka (2002).

The consequence is that if we use the principle of marginalization without further model combination techniques, each model has to have a high complexity to cope with the variations in the data. An example for such a model class are non-parametric models presented in Section 2.4.

2.2.3 Model Combination with Bagging

A common model combination technique proposed by Breiman (1996) is *bootstrap aggregating (Bagging)* and can be applied to regression or classification tasks. It belongs to a much wider category of learning approaches called ensemble methods, which try to combine multiple simple models often referred to as *base classifiers* or *weak classifiers*. The idea of Bagging is to learn several models $(\boldsymbol{\theta}_t)_{t=1}^T$ by using only a random fraction r_B of all available training data for each model estimation or training. The final combination is done by averaging all model responses:

$$h^{\text{Bagging}}(\boldsymbol{x}_*) \stackrel{\text{def}}{=} \frac{1}{T}\sum_{t=1}^{T} h(\boldsymbol{x}_*;\boldsymbol{\theta}_t) \ . \tag{2.17}$$

Due to the use of multiple and diverse models, Bagging is able to reduce overfitting effects and is thus especially suitable for classification models that have a high complexity and a high resulting variance with respect to the training set. Let us take a closer look on the generalization properties of this estimator by analyzing the error made by the predictor when trying to estimate the underlying ground truth function $\tilde{h}$ (Bishop, 2006, Section 14.2). The output of each model can be written using error terms ε_t:

$$h(\boldsymbol{x};\boldsymbol{\theta}_t) = \tilde{h}(\boldsymbol{x}) + \varepsilon_t(\boldsymbol{x}) \ . \tag{2.18}$$

The error of a single model is given by the expected squared difference to the ground truth function:

$$err_t = \mathbb{E}_{\boldsymbol{x}}\left(\left(h(\boldsymbol{x};\boldsymbol{\theta}_t) - \tilde{h}(\boldsymbol{x})\right)^2\right) = \mathbb{E}_{\boldsymbol{x}}\left(\varepsilon_t(\boldsymbol{x})^2\right) \ , \tag{2.19}$$

which directly leads to the average error of the ensemble when each model is used separately:

$$err_{\text{average}} = \frac{1}{T}\sum_{t=1}^{T} err_t = \frac{1}{T}\sum_{t=1}^{T} \mathbb{E}_{\boldsymbol{x}}\left(\varepsilon_t(\boldsymbol{x})^2\right) \quad . \tag{2.20}$$

Analyzing the error of Bagging is now straightforward and we only assume that the error terms do not correlate and have a zero mean:

$$\begin{aligned} err_{\text{bagging}} &= \mathbb{E}_{\boldsymbol{x}}\left(\left(h_{\mathcal{D}}^{\text{Bagging}}(\boldsymbol{x}_*) - \tilde{h}(\boldsymbol{x})\right)^2\right) \\ &= \mathbb{E}_{\boldsymbol{x}}\left(\left(\frac{1}{T}\sum_{t=1}^{T}\varepsilon_t(\boldsymbol{x})\right)^2\right) \\ &= \frac{1}{T^2}\left(\mathbb{E}_{\boldsymbol{x}}\left(\varepsilon_1(\boldsymbol{x})^2 + \ldots + \varepsilon_T(\boldsymbol{x})^2\right) + \mathbb{E}_{\boldsymbol{x}}\left(\sum_{t\neq t'}\varepsilon_t(\boldsymbol{x})\varepsilon_{t'}(\boldsymbol{x})\right)\right) \\ &\overset{*}{=} \frac{1}{T^2}\left(\mathbb{E}_{\boldsymbol{x}}\left(\varepsilon_1(\boldsymbol{x})^2 + \ldots + \varepsilon_T(\boldsymbol{x})^2\right)\right) \\ &= \frac{1}{T}\cdot err_{\text{average}} \quad . \end{aligned} \tag{2.21}$$

Hence, in theory Bagging helps to reduce the expected error, especially when using a large size of the ensemble. The important assumption we made in the previous derivations is that the error terms are uncorrelated. In general, this assumption does not hold and is even impossible to achieve. Bagging tries to build an ensemble of independent models by training each of them with a different, although probably overlapping, training set. The randomization used by Bagging is thus justified as a simple method to reduce the correlation between base classifiers of an ensemble. For probabilistic models, Bagging can also be applied to average the predictive posterior distributions of a label y_* corresponding to an unseen example $\boldsymbol{x}_*$:

$$p(y_* \mid \boldsymbol{x}_*, \mathcal{D}) = \frac{1}{T}\sum_{t=1}^{T} p(y_* \mid \boldsymbol{x}_*, \boldsymbol{\theta}_t) \quad . \tag{2.22}$$

Note that this does not directly correspond to the probabilistic version of Eq. (2.17).

2.3 Random Decision Forests

The following section reviews *random decision forest (RDF)* as an efficient classification method. First, we describe the basic principles of decision trees (Breiman et al., 1984), one of the fundamentals of early machine learning, and illustrate how current versions make use of Bagging and randomized learning.

2.3.1 Decision Trees

Decision tree classifiers can be regarded as model combination techniques. The underlying idea of their use for classification purposes is that the input space is recursively split by binary base classifiers leading to a binary tree structure. Each inner node of the tree corresponds to such a classifier $h : \mathbb{R}^D \to \{-1, 1\}$. Although arbitrary classification models can be used, in this thesis we stick to axis-aligned hyperplanes parameterized with the index $r \in \{1, \dots, D\}$ of the feature and a threshold $\zeta \in \mathbb{R}$:

$$h(\boldsymbol{x}; r, \zeta) = \operatorname{sign}(x_r - \zeta) = \begin{cases} 1 & \text{if } x_r > \zeta \\ -1 & \text{if } x_r \leq \zeta \end{cases} . \tag{2.23}$$

Base classifiers determine the path of an example $\boldsymbol{x}$ within the tree, which ends in a leaf node $\vartheta(\boldsymbol{x})$. Thus, the input space is split into several regions, each of them corresponding to one of the m_ℓ leaves. Each leaf node is associated with a posterior distribution $p(y = k \mid \vartheta(\boldsymbol{x}))$, which is an estimate of the probability of $\boldsymbol{x}$ belonging to class k given that this specific leaf is reached. Therefore, a tree is theoretically able to approximate arbitrary class regions. The general principles and terms are illustrated in Figure 2.2.

One of the advantages of decision trees is the few number of base classifiers which have to be used to classify a new example. If the tree is balanced, we only have to apply $\mathcal{O}(\log m_\ell)$ binary base classifiers to get an approximation of the posterior.

2.3.1.1 Learning of Decision Trees

Learning a decision tree requires building the binary tree structure and estimating posterior probabilities. Finding an optimal decision tree for a given training dataset $\mathcal{D}$ is a combinatorial optimization problem and intractable to solve. However, a greedy strategy is straightforward to develop. We start from the root

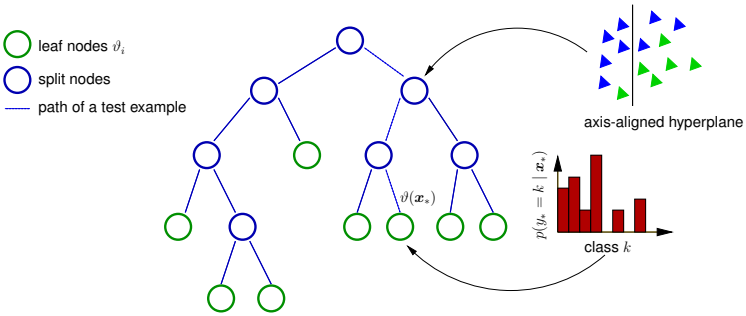


Figure 2.2: General principle and terms of decision trees. The right diagram illustrates the posterior distribution corresponding to a leaf node. The path of an example in the tree (dashed line) is determined by base classifiers related to each split node.

node and choose the base classifier that best splits the training data according to some criterion $\mathit{\Gamma}$. In each child node, this procedure is recursively continued with the reduced training data, until a termination criteria is fulfilled.

A split criterion is mostly based on an impurity measure $\mathcal{J}$, which returns high values if examples of several different classes reached the current node. One common measure is the entropy of the labels:

$$\mathcal{J}^{\text{Entropy}}(v) \stackrel{\text{def}}{=} -\sum_{k=1}^{M} p_k \ \log \ p_k \ , \tag{2.24}$$

with p_k being the ratio of examples of class k in the current node v. The final split criterion is defined as the expected reduction of the impurity measure if using the current base classifier:

$$\mathit{\Gamma}(v; r, \zeta) \stackrel{\text{def}}{=} \mathcal{J}(v) - \sum_{v' \text{ is a child of } v} p(v' \mid v) \ \mathcal{J}(v') \ . \tag{2.25}$$

The summation is done over every child node which is generated by the base classifier $h(\cdot; r, \zeta)$ and $p(v'|v)$ denotes the transition probability that an input example reaches node v' from v. The transition probability can be easily estimated by counting the number of examples reaching each child node. In case of $\mathcal{J}$ being the entropy, $\mathit{\Gamma}$ is equivalent to the mutual information of the label

and the output of the base classifier. Optimization of the objective function in Eq. (2.25) with respect to the parameters of the axis-aligned base classifiers is done in traditional decision tree approaches with an exhaustive search over all possible thresholds ζ and features $r \in \{1, \ldots, D\}$ (components of a feature vector):

$$\left(\hat{r}_v, \hat{\zeta}_v\right) = \underset{1 \leq r \leq D, \zeta \in \mathbb{R}}{\operatorname{argmax}} \; \Gamma(v; r, \zeta) \;\; . \tag{2.26}$$

Important ingredients of the learning procedure are the termination criteria. A trivial case appears when a node, after recursive splitting of the training set, only contains examples of a single class. This condition can be relaxed by thresholding the impurity measure of the current node, *i.e.* if $\mathcal{J}(v)$ falls below a certain value $\xi_{\mathcal{J}}$. Another criterion is a lower bound ξ_{n} of the number of examples, which is reasonable because it is difficult to learn a good suitable base classifier without a sufficient size of the training set (*cf.* the motivation of this thesis in Section 1.1). To restrict the worst-case performance of the classifier during classification, the maximum depth of the tree can additionally be bounded by a parameter ξ_{d}.

One severe disadvantage of decision trees is their tendency to overfitting, which results from the high flexibility of the model and the weak threshold-based regularization using termination criteria. To overcome this problem, heaps of so called *pruning* techniques have been proposed (Breiman et al., 1984), which mostly utilize an additional validation set to guide the shrinkage of the tree after applying the traditional learning scheme.

2.3.2 Randomized Learning of Decision Forests

Randomized learning offers to reduce overfitting effects efficiently without using some heuristic pruning approach. In general, there are several parts of the learning process in which randomization can be introduced.

As mentioned earlier in Section 2.2.3, Bagging can reduce overfitting effects, which makes it perfectly suitable for the combination of decision trees (Breiman, 2001). Several trees are learned with a random fraction r_{B} of the training data resulting in an ensemble often referred to as *random forest*. We denote the leaf node of tree t reached by an example $\boldsymbol{x}$ with $\vartheta^{(t)}(\boldsymbol{x})$. Instead of using a single estimate of the posterior probability of the label, the estimates of all trees can be

used and averaged like done in the Bagging framework (*cf.* Eq. (2.22)):

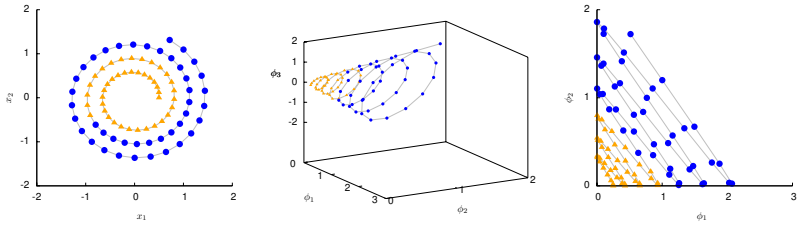
$$p(y_* \mid \boldsymbol{x}_*, \mathcal{D}) = \frac{1}{T} \sum_{t=1}^{T} p(y_* \mid \boldsymbol{x}_*, \vartheta^{(t)}(\boldsymbol{x}_*)) \ . \tag{2.27}$$

The parameter r_{B} has to be carefully chosen. On the one hand, learning a decision tree with only a small number of examples leads to completely misleading classification models. On the other hand, a smaller number of examples for each tree decreases the computation time of learning and increases the diversity within the ensemble. An additional important advantage of Bagging is the availability of out-of-sample estimates. The idea is that every training example $\boldsymbol{x}$ is classified by the reduced ensemble of all decision trees not trained with $\boldsymbol{x}$. The resulting classification results are then used to measure the expected accuracy, which allows for optional model selection.

A similar approach is the random subspace approach of Ho (1998). Each tree is trained using the full training set but with only a fraction of all available features, *i.e.* selections of input dimensions. In contrast, Kuncheva and Rodríguez (2007) propose a method called *rotation forests*, which generalizes the work of Ho (1998) by building each tree with randomly linear transformed features. Instead of restricting the set of features for a whole tree, Geurts et al. (2006) suggest randomizing the optimization of base classifiers during learning. Whereas traditional decision tree approaches exhaustively search for the best suitable pair of feature and threshold, the algorithm of Geurts et al. (2006) randomly selects a subset $\mathcal{R}_v$ of all features and a set Q_v of thresholds in each inner node v and restricts the search for the best suitable base classifier to the resulting reduced hypothesis space:

$$\left(\hat{r}_v, \hat{\zeta}_v\right) = \operatorname*{argmax}_{r \in \mathcal{R}_v, \zeta \in Q_v} \Gamma(v; r, \zeta) \ . \tag{2.28}$$

This strategy leads to a significant speed-up of learning controlled by the size of the sets $\mathcal{R}_v$ and Q_v. Shotton et al. (2008) show the impressive performance of randomized learning in a semantic segmentation application, which needs to handle millions of training examples. In this thesis, we use a combination of the ideas presented by Breiman (2001) and Geurts et al. (2006) and refer to it as *random decision forest*.



(a) input space $\mathcal{X}$ (b) feature space $\mathcal{H}$ (c) first two components of $\mathcal{H}$

Figure 2.3: Illustration of the usage of kernels: (a) input space, (b) the corresponding feature space using a polynomial kernel and (c) the first two components of the feature space showing that the data can now be linearly separated. Points are connected by lines to visualize the data structure and to generate a correspondence between the plots.

2.4 Learning with Kernels

As we have observed in Section 1.1, visual object recognition requires highly flexible classification models. Despite the success of randomized decision forests introduced in the last section, they are not able to represent nonlinear decision boundaries in an efficient manner. In contrast, the usage of kernels in powerful classifiers, like support vector machines and Gaussian processes (Section 2.5 and Section 2.6), offer to model nonlinearity without requiring additional parameterization.

2.4.1 Kernels and High-dimensional Feature Spaces

A large number of classifiers assume that the input data of the classification problem is linearly separable try to find a suitable hyperplane separating the data. Such methods are referred to as *linear classifiers* even though they often include a bias term to model hyperplanes that do not necessarily pass through the origin. Figure 2.3(a) shows a simple synthetic example which can not be solved by a linear classifier (Schölkopf and Smola, 2001, Section 2). In the following, we use the term *nonlinear problem* for such tasks. The illustrated example can be solved by using an explicit nonlinear classification model. For example, we

could estimate a circle or ellipse separating the data:

$$h(\boldsymbol{x}) = \text{sign}\left(w_1 x_1^2 + w_2 x_1 x_2 + w_3 x_2^2 + b\right) \tag{2.29}$$

$$= \begin{cases} 1 & \text{if } w_1 x_1^2 + w_2 x_1 x_2 + w_3 x_2^2 > -b \\ -1 & \text{otherwise} \end{cases} . \tag{2.30}$$

However, we can also regard this classifier as a linear one applied to nonlinear transformed input vectors $\phi(\boldsymbol{x})$:

$$\phi(\boldsymbol{x}) = \left(x_1^2,\, x_1 \cdot x_2,\, x_2^2\right)^T \quad , \tag{2.31}$$

$$h(\boldsymbol{x}) = \text{sign}\left(\langle \boldsymbol{w}, \phi(\boldsymbol{x}) \rangle + b\right) \quad , \tag{2.32}$$

where $\langle \cdot, \cdot \rangle$ denotes a scalar product. The result of the transformation ϕ is visualized in the center 3d plot of Figure 2.3. Figure 2.3(c) shows a 2d projection of the transformed data highlighting the fact that the transformation leads to linearly separable data. With D-dimensional input vectors $\boldsymbol{x} \in \mathbb{R}^D$ the transformation would involve computing $\mathcal{O}(D^2)$ terms, which is impractical and does not generalize well to higher-order problems. Kernel functions allow us to skip explicit computation and even specification of ϕ.

The generalized *representer theorem* of Schölkopf and Smola (2001) offers for many learning approaches, such as support vector machines (Section 2.5) or Gaussian process regression (Section 2.6.5), the possibility of writing the hyperplane $\boldsymbol{w}$ in terms of learning examples $\boldsymbol{x}_i$ (Section A.1):

$$\boldsymbol{w} = \sum_{i=1}^{n} \alpha_i \phi(\boldsymbol{x}_i) \quad , \tag{2.33}$$

with coefficients $\alpha_i \in \mathbb{R}$. As can be seen in Eq. (2.32), an important ingredient of the hypothesis is the evaluation of the scalar product $\langle \boldsymbol{w}, \phi(\boldsymbol{x}) \rangle$, which can be expressed using Eq. (2.33) as follows:

$$\langle \boldsymbol{w}, \phi(\boldsymbol{x}) \rangle = \sum_{i=1}^{n} \alpha_i \langle \phi(\boldsymbol{x}_i), \phi(\boldsymbol{x}) \rangle \quad , \tag{2.34}$$

where we utilized the bilinearity properties of a scalar product. Thus, evaluating the hypothesis is merely based on calculating the scalar product in $\mathcal{H}$ induced by $\phi : \mathcal{X} \to \mathcal{H}$. This space is referred to as *feature space*. The feature space $\mathcal{H}$

has to be equipped with a scalar product, which is the fact for so called *Hilbert spaces*. In the remainder of this thesis, we use the notation $\langle \cdot, \cdot \rangle_{\mathcal{H}}$ to explicitly refer to the scalar product in $\mathcal{H}$. If we can find a *kernel* or *kernel function* $K : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ that computes the scalar product in the high-dimensional or even infinite-dimensional space $\mathcal{H}$, *i.e.*

$$\forall \boldsymbol{x}, \boldsymbol{x}' \in \mathcal{X} \; : \; K(\boldsymbol{x}, \boldsymbol{x}') = \; \langle \phi(\boldsymbol{x}), \phi(\boldsymbol{x}') \rangle_{\mathcal{H}} \quad , \tag{2.35}$$

we do not have to specify the feature map ϕ in an explicit manner and the kernel function can be used as a substitute for the scalar product without even knowing anything about a corresponding transformation. This technique is known as the *kernel trick* and one of the most important advantages is the possibility to handle structured data or input data with different dimensionality. Thus, the input data could consist of images $\boldsymbol{x}$ with different sizes and the only thing which has to be done to perform learning is to compare them in a suitable way (Section 4.3).

The question remains how do we know that a given function K is a suitable kernel function. Of course, one way is always to construct a corresponding ϕ, but this can be considered as a difficult mathematical undertaking and can be circumvented by the Mercer condition presented in the next section.

2.4.2 Mercer Condition and Construction of Kernels

The Mercer condition is one of the fundamentals in machine learning with kernels. It allows treating ϕ as a mysterious black box rather than as a transformation that has to be well designed. Let us first give a definition of a special and important class of kernels.

Definition 2.1 (Positive definite kernel) *A function $K : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ is called a* positive definite kernel *(function), if for all $n \in \mathbb{N}$ and all finite subsets $\mathbf{X} = (\boldsymbol{x}_i)_{i=1}^n$ of $\mathcal{X}$ the quadratic matrix $\mathbf{K} \in \mathbb{R}^{n \times n}$ with $K_{ij} = K(\boldsymbol{x}_i, \boldsymbol{x}_j)$ is positive semi-definite. Such a matrix $\mathbf{K}$ is called* kernel matrix.

We use the notation $K(\mathbf{X}, \mathbf{X}')$ with $\mathbf{X}' = (\boldsymbol{x}'_j)_{j=1}^m$ to refer to the $n \times m$ matrix $\mathbf{A}$ containing the pairwise kernel values $A_{ij} = K(\boldsymbol{x}_i, \boldsymbol{x}'_j)$. If $\mathbf{X}'$ only contains one element $\boldsymbol{x}$, we also write $K(\mathbf{X}, \boldsymbol{x})$.

This definition directly leads to a simple fact about kernel combination, which is needed as an essential tool in Section 2.6.10.

Proposition 2.2 (Linear kernel combination) *If $K_1, \ldots, K_R$ are positive definite kernels (kernel functions) and $\beta_1, \ldots, \beta_R \in \mathbb{R}$ are non-negative coefficients, the linear combination $K = \beta_1 K_1 + \ldots + \beta_R K_R$ is also positive definite.*

The proposition shows that we can scale positive definite kernel functions with $v_0 > 0$ and add a non-negative bias v_1 without violating their positive definiteness:

$$\tilde{K}(\cdot, \cdot) = v_0 \cdot K(\cdot, \cdot) + v_1 \quad . \tag{2.36}$$

If we are using a linear kernel $K(\boldsymbol{x}, \boldsymbol{x}') = \langle \boldsymbol{x}, \boldsymbol{x}' \rangle$, *i.e.* the feature map ϕ is the identity, it can be seen that every kernel matrix induced by K is positive semi-definite. This suggests a strong relationship between condition (2.35) and definition 2.1, which can be made explicit with the Mercer condition.

Theorem 2.3 (Mercer condition) *(Mercer, 1909; Vapnik, 2000)*
A function $K : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ is a positive definite kernel if and only if there exists a Hilbert space $\mathcal{H}$ and a feature map $\phi : \mathcal{X} \to \mathcal{H}$ such that

$$\forall \boldsymbol{x}, \boldsymbol{x}' \in \mathcal{X} \ : \ K(\boldsymbol{x}, \boldsymbol{x}') = \langle \phi(\boldsymbol{x}), \phi(\boldsymbol{x}') \rangle_{\mathcal{H}} \quad . \tag{2.37}$$

Proof:

"$\rightarrow$":

Let us assume that we have a finite input space $\mathcal{X}$ with $|\mathcal{X}| = n$. In the following, we only give a proof for this special case, since it allows us to use well-known concepts of linear algebra instead of their corresponding generalizations for functions. An exact proof can be found in Schölkopf and Smola (2001). If we have a positive definite kernel, the kernel matrix $\mathbf{K}$ of all elements of the input space is positive semi-definite. Thus, $\mathbf{K}$ can be decomposed using the Cholesky decomposition by $\mathbf{K} = \mathbf{G}\mathbf{G}^T$. If we define $\phi(\boldsymbol{x}_i)$ to be the column vector consisting of the elements of the i'th row of $\mathbf{G}$, we immediately see that ϕ obeys Eq. (2.37).

"$\leftarrow$":

Let $\mathbf{K}$ be an arbitrary kernel matrix created by K obeying Eq. (2.37). For every

$\boldsymbol{\alpha} \in \mathbb{R}^n$ the following holds:

$$\boldsymbol{\alpha}^T \mathbf{K} \boldsymbol{\alpha} = \sum_{i,j=1}^{n} \alpha_i \alpha_j \langle \phi(\boldsymbol{x}_i), \phi(\boldsymbol{x}_j) \rangle_{\mathcal{H}} = \left\langle \sum_{i=1}^{n} \alpha_i \phi(\boldsymbol{x}_i), \sum_{j=1}^{n} \alpha_j \phi(\boldsymbol{x}_j) \right\rangle_{\mathcal{H}}$$
$$= \left\| \sum_{i=1}^{n} \alpha_i \phi(\boldsymbol{x}_i) \right\|_{\mathcal{H}}^2 \geq 0 \ , \tag{2.38}$$

with $||\cdot||_{\mathcal{H}}$ being the induced norm in $\mathcal{H}$. Thus, $\mathbf{K}$ is positive semi-definite and due to the arbitrary selection of this matrix it follows that the function K is a positive definite kernel. □

In the remainder of this thesis, we use the terms *kernel* or *kernel function* to refer to positive definite kernels. A kernel function can be regarded as a special type of similarity measure between two inputs. Therefore, the kernel trick leads to a machine learning algorithm that solely learns using the similarity of given inputs rather than their explicit representation. Let us now consider some kernel functions that are widely used in a large set of machine learning applications and that are utilized in this thesis.

Definition 2.4 (Families of kernel functions) *A kernel function $K : \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}$ is called*

1. stationary *or* shift-invariant *if it solely depends on the difference between both input vectors:*

$$K(\boldsymbol{x}, \boldsymbol{x}') = K^{stat}(\boldsymbol{x} - \boldsymbol{x}') \ , \tag{2.39}$$

2. homogeneous *or* generalized radial basis function kernel*, if it solely depends on the distance between both input vectors measured by some metric $d : \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}$*

$$K(\boldsymbol{x}, \boldsymbol{x}') = K^{hom}(d(\boldsymbol{x}, \boldsymbol{x}')) \tag{2.40}$$

3. radial basis function (rbf) *kernel, if it is homogeneous and solely depends on the Euclidean distance of both input vectors* [2].

[2] Bishop (2006) uses a more general notion of the term radial basis function kernel defined as a synonym for homogeneous kernels.

One of the most widely applied kernel functions is the Gaussian kernel, which belongs to the family of radial basis function kernels:

$$K^{\text{gauss}}(\boldsymbol{x}, \boldsymbol{x}') \stackrel{\text{def}}{=} \exp\left(-\gamma \left|\left|\boldsymbol{x} - \boldsymbol{x}'\right|\right|^2\right) \quad . \tag{2.41}$$

The kernel is parameterized by $\gamma > 0$, which can have an important influence on the behavior of the machine learning algorithm. For example, $\gamma = 0$ leads to uniform kernel values independent of the input data, whereas $\gamma \to \infty$ approaches to the delta function. Parameters of kernel functions are termed *hyperparameters* and their estimation or optimization with respect to a specific task is an important step and described for support vector machines or Gaussian process based methods in Section 2.5.5 and Section 2.6.10, respectively.

Non-stationary kernels have shown to be useful when comparing histograms (Grauman and Darrell, 2007), such as the ones calculated by the bag of visual words approach presented in Section 4.2. The two most prominent non-stationary kernels in this area are the *chi-square kernel* and the *minimum intersection kernel*:

$$K^{\chi^2}(\boldsymbol{x}, \boldsymbol{x}') \stackrel{\text{def}}{=} \exp\left(-\gamma \sum_{i=1}^{D} \frac{(x_i - x_i')^2}{x_i + x_i'}\right) \quad , \tag{2.42}$$

$$K^{\text{min}}(\boldsymbol{x}, \boldsymbol{x}') \stackrel{\text{def}}{=} \sum_{i=1}^{D} \min(x_i, x_i') \quad . \tag{2.43}$$

Choosing a kernel function for a specific task can be tricky. It is always advantageous to incorporate as much prior knowledge about the application as possible. If such a prior knowledge is not directly available, we can simply use the Gaussian kernel as a default choice or rely on model selection techniques, such as those described in Rasmussen and Williams (2005) for GP based algorithms and reviewed in Section 2.6.10.

2.5 Support Vector Machines

Nowadays, support vector machines (SVM) are a common classification technique in a large set of computer vision related tasks; *e.g.* pedestrian detection (Dalal and Triggs, 2005), object localization (Felzenszwalb et al., 2008), action recognition (Ullah et al., 2010), high-level object recognition (Farhadi et al., 2009). The SVM approach is linear in its original version but can be

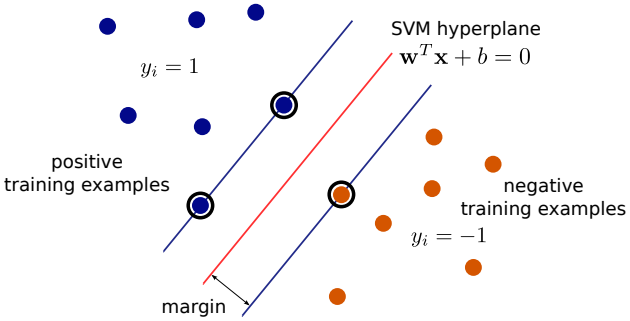


Figure 2.4: Basic principle of margin maximization with support vector machines. Colored points represent training examples and are additionally highlighted if they are support vectors.

extended to nonlinear problems by applying the previously presented kernel trick in the dual version of the underlying optimization problem. We restrict ourselves to a description of aspects necessary for this thesis and we refer to Schölkopf and Smola (2001) and Bishop (2006) for a comprehensive treatment of this topic.

2.5.1 Binary Classification with SVM

Let us start with binary classification tasks and labels $y \in \{-1, 1\}$. The aim of SVM is to separate two classes in a linear manner using a hyperplane $\boldsymbol{w} \in \mathbb{R}^D$ represented by the function:

$$f(\boldsymbol{x}) = \langle \boldsymbol{w}, \boldsymbol{x} \rangle + b \ . \tag{2.44}$$

In the following, we assume the training set to be linearly separable. The characteristic property of SVM is the determination of a hyperplane that separates the training data and maximizes the *margin* $\text{mg}_{\mathcal{D}}(\boldsymbol{w}, b)$ defined as the smallest distance of the hyperplane to a training example (Bishop, 2006, Section 7.1, p. 326). Data vectors $\boldsymbol{x}_i$ having the smallest distance to the hyperplane are termed *support vectors*, because they determine the hyperplane. These principles and terms are visualized in Figure 2.4. The resulting optimization problem can be

written as follows:

$$\begin{aligned} &\underset{\boldsymbol{w}\in\mathbb{R}^D,\, b\in\mathbb{R}}{\text{maximize}} && \text{mg}_{\mathcal{D}}(\boldsymbol{w}, b) \\ &\text{subject to} && \forall i = 1 \ldots n : \; y_i \cdot f(\boldsymbol{x}_i) > 0 \;\; . \end{aligned} \tag{2.45}$$

The inequalities ensure that the hyperplane separates the training set. Given a training set $\mathcal{D} = \{(\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_n, y_n)\}$ the margin can be expressed by:

$$\text{mg}_{\mathcal{D}}(\boldsymbol{w}, b) = \min \{||\boldsymbol{x} - \boldsymbol{x}_i|| : 1 \leq i \leq n; \langle \boldsymbol{w}, \boldsymbol{x} \rangle + b = 0; \boldsymbol{x} \in \mathcal{X}\} \tag{2.46}$$

$$= \min_{i=1\ldots n} \frac{|\, \langle \boldsymbol{w}, \boldsymbol{x}_i \rangle + b \,|}{||\boldsymbol{w}||} \;\; . \tag{2.47}$$

Scaling the hyperplane vector $\boldsymbol{w}$ and the bias b with $\lambda > 1$ does not lead to a violation of the constraints or a change of the margin. Due to this reason, the scaling can be selected arbitrarily and we fix it such that $|\, \langle \boldsymbol{w}, \boldsymbol{x}_i \rangle + b \,| = 1$ holds for every $\boldsymbol{x}_i$ having minimal distance to the hyperplane. The advantage is the simplification of the margin to $\text{mg}_{\mathcal{D}}(\boldsymbol{w}, b) = 1/\,||\boldsymbol{w}||$. We can now rewrite the SVM optimization problem as:

$$\begin{aligned} &\underset{\boldsymbol{w}\in\mathbb{R}^D,\, b\in\mathbb{R}}{\text{minimize}} && \frac{1}{2}\,||\boldsymbol{w}||^2 \\ &\text{subject to} && \forall i = 1 \ldots n : \; y_i \cdot (\langle \boldsymbol{w}, \boldsymbol{x}_i \rangle + b) \geq 1 \;\; . \end{aligned} \tag{2.48}$$

The additional constant factor $\frac{1}{2}$ is introduced due to mathematical convenience in later derivations. Note that the right hand side of our constraints changed from > 0 to ≥ 1, which can be explained by considering the old constraint $y_i \cdot (\langle \boldsymbol{w}, \boldsymbol{x}_i \rangle + b) > 0$ and incorporating the definition of the scaling of $\boldsymbol{w}$ and b:

$$y_i \cdot (\langle \boldsymbol{w}, \boldsymbol{x}_i \rangle + b) = |y_i \cdot (\langle \boldsymbol{w}, \boldsymbol{x}_i \rangle + b)\,| = |y_i| \cdot |\,(\langle \boldsymbol{w}, \boldsymbol{x}_i \rangle + b)\,| \tag{2.49}$$

$$= |\,(\langle \boldsymbol{w}, \boldsymbol{x}_i \rangle + b)\,| \geq 1 \;\; . \tag{2.50}$$

The SVM optimization problem is a quadratic program and several efficient solvers exist, which are partly described in (Schölkopf and Smola, 2001).

2.5.2 Soft Margin Classifier

The problem remains that linearly separable training data are unlikely, given a low dimension of the input space and we have to handle misclassifications

during learning, *i.e.* hyperplanes that are not necessarily separating the learning examples. The key idea to tackle this problem is the introduction of *slack variables* $\xi_i \geq 0$ (Bishop, 2006) and relaxing the conditions in Eq. (2.48) to:

$$y_i \cdot (\langle \boldsymbol{w}, \boldsymbol{x}_i \rangle + b) \geq 1 - \xi_i \quad . \tag{2.51}$$

Violating the conditions with $\xi_i > 0$ is penalized by an additional weighted term which results in the following, modified optimization problem often referred to as *soft margin classifier* (Schölkopf and Smola, 2001, p. 16)

$$\begin{aligned} \underset{\boldsymbol{w}\in\mathbb{R}^D,\, b\in\mathbb{R},\, \boldsymbol{\xi}\in\mathbb{R}^n}{\text{minimize}} \quad & \frac{1}{2}\left|\left|\boldsymbol{w}\right|\right|^2 + C\sum_{i=1}^{n}\xi_i \\ \text{subject to} \quad & \forall i:\ y_i \cdot (\langle \boldsymbol{w}, \boldsymbol{x}_i \rangle + b) \geq 1 - \xi_i \ \text{ and } \ \xi_i \geq 0 \ . \end{aligned} \tag{2.52}$$

The parameter $C > 0$ is used to tune the trade-off between a large margin and allowed misclassifications during training.

Another formulation of the optimization problem offers some further insights into SVM classification and is also essential to establish some important relationships to other classification techniques as done in Section 2.6. First of all, we need to define the *hinge loss* H:

$$H(z) = \max(1 - z, 0) \quad . \tag{2.53}$$

Using the hinge loss, we can express the previous optimization problem (2.52) as a single objective function without constraints and with the regularization parameter $\lambda = (2C)^{-1}$ (Bishop, 2006, p. 337):

$$\underset{\boldsymbol{w}\in\mathbb{R}^D, b\in\mathbb{R}}{\text{minimize}} \quad \sum_{i=1}^{n} H\left(y_i \cdot f(\boldsymbol{x}_i)\right) + \ \lambda\left|\left|\boldsymbol{w}\right|\right|^2 \quad . \tag{2.54}$$

The hinge loss can be regarded as an approximation of the misclassification error (see Section 2.6.5 for further details). Thus, the SVM approach minimizes a regularized risk functional consisting of an error term and a complexity measure of the resulting decision function. Note that this scheme is connected to maximum a posteriori estimation (Section 2.2.1) with the error term corresponding to an improper negative logarithmic likelihood and $\left|\left|\boldsymbol{w}\right|\right|^2$ serving as the negative logarithm of a Gaussian prior on $\boldsymbol{w}$.

2.5.3 Duality and Kernels

In Section 2.5.1, we described support vector machines as a maximum margin classification technique and also tackled the problem of nonlinearly separable data using slack variables. However, the resulting classifier is still linear and not able to provide complex decision boundaries as demanded by visual recognition tasks. As described in the last section, a solution is the integration of the kernel trick into the learning algorithm. In the following, we consider the SVM approach applied to input examples transformed with a map $\phi : \mathcal{X} \rightarrow \mathcal{H}$:

$$f(\boldsymbol{x}) = \langle \boldsymbol{w}, \phi(\boldsymbol{x}) \rangle_{\mathcal{H}} + b \ . \tag{2.55}$$

We already mentioned the representer theorem of Schölkopf and Smola (2001) given in Section A.1, which states that we are able to write the hyperplane normal $\boldsymbol{w}$ in terms of the transformed training examples (*cf.* Eq. (2.33)) for various kinds of optimization problems, such as those related to the SVM approach. However, we can show this important result directly for the SVM approach by considering the corresponding dual optimization problem of the primal problem (2.48). We derive the dual problem using the corresponding Lagrangian:

$$L(\boldsymbol{w}, b, \boldsymbol{\lambda}) = \frac{1}{2} ||\boldsymbol{w}||^2 + \sum_{i=1}^{n} \lambda_i \left(1 - y_i \cdot (\langle \boldsymbol{w}, \phi(\boldsymbol{x}_i) \rangle_{\mathcal{H}} + b)\right) \ . \tag{2.56}$$

To calculate the Lagrange dual function, we have to minimize L with respect to $\boldsymbol{w}$ and b (Boyd and Vandenberghe, 2004, chapter 5), which involves calculating the gradients:

$$\{\nabla_{\boldsymbol{w}} L\}(\boldsymbol{w}, b, \boldsymbol{\lambda}) = \boldsymbol{w} - \sum_{i=1}^{n} \lambda_i y_i \left(\phi(\boldsymbol{x}_i)\right) \ , \tag{2.57}$$

$$\{\nabla_b L\}(\boldsymbol{w}, b, \boldsymbol{\lambda}) = -\sum_{i=1}^{n} \lambda_i y_i = -\boldsymbol{\lambda}^T \boldsymbol{y} \ . \tag{2.58}$$

Setting both gradients to zero we arrive at:

$$\tilde{\boldsymbol{w}} = \sum_{i=1}^{n} \lambda_i y_i \phi(\boldsymbol{x}_i) = \sum_{i=1}^{n} \alpha_i \phi(\boldsymbol{x}_i) \ , \tag{2.59}$$

with coefficients $\alpha_i = \lambda_i y_i$ and the property $\sum_{i=1}^{n} \alpha_i = 0$. The result is an instance of the representer theorem for SVM based learning. Note that a coefficient α_i is zero if and only if the corresponding learning example is not a support vector, *i.e.* the i-th constraint in (2.48) is not active. The norm of $\tilde{\boldsymbol{w}}$ can be written in terms of kernel evaluations as:

$$||\tilde{\boldsymbol{w}}||^2 = \sum_{i,j=1}^{n} \alpha_i \alpha_j \left\langle \phi(\boldsymbol{x}_i), \phi(\boldsymbol{x}_j) \right\rangle_{\mathcal{H}} = \boldsymbol{\alpha}^T \mathbf{K} \boldsymbol{\alpha} \ . \tag{2.60}$$

We can now combine everything to determine a modified Lagrange dual function $g(\boldsymbol{\alpha}) = L\left(\tilde{\boldsymbol{w}}, \tilde{b}, [\alpha_i / y_i]_{i=1}^{n}\right)$, which depends on the coefficients α_i rather than on λ_i:

$$\begin{aligned}
g(\boldsymbol{\alpha}) &= \frac{1}{2}\boldsymbol{\alpha}^T \mathbf{K} \boldsymbol{\alpha} + \sum_{i=1}^{n} \left(\frac{\alpha_i}{y_i}\right) \left(1 - y_i \cdot \left(\sum_{j=1}^{n} \alpha_j \left\langle \phi(\boldsymbol{x}_j), \phi(\boldsymbol{x}_i) \right\rangle_{\mathcal{H}} + \tilde{b}\right)\right) \\
&= \frac{1}{2}\boldsymbol{\alpha}^T \mathbf{K} \boldsymbol{\alpha} + \left(\sum_{i=1}^{n} \frac{\alpha_i}{y_i}\right) - \left(\sum_{i,j=1}^{n} \alpha_i \alpha_j \left\langle \phi(\boldsymbol{x}_j), \phi(\boldsymbol{x}_i) \right\rangle_{\mathcal{H}}\right) \\
&\quad - \tilde{b}\left(\sum_{i=1}^{n} \alpha_i\right) \\
&= -\frac{1}{2}\boldsymbol{\alpha}^T \mathbf{K} \boldsymbol{\alpha} + \sum_{i=1}^{n} \underbrace{\left(\frac{\alpha_i}{y_i}\right)}_{=\alpha_i y_i} \\
&= -\frac{1}{2}\boldsymbol{\alpha}^T \mathbf{K} \boldsymbol{\alpha} + \boldsymbol{\alpha}^T \boldsymbol{y} \ .
\end{aligned} \tag{2.61}$$

We use the kernel matrix $\mathbf{K}$ of the training set that arises from the pairwise scalar products of the transformed learning examples, *i.e.* $K_{ij} = \left\langle \phi(\boldsymbol{x}_i), \phi(\boldsymbol{x}_j) \right\rangle_{\mathcal{H}}$. The (modified) dual problem maximizes $g(\boldsymbol{\alpha})$ with respect to constraints resulting from the non-negativeness of the Lagrange multipliers λ_i and the fact that the coefficients α_i sum to zero, which was derived by setting the gradient of b

(Eq. (2.58)) to zero:

$$\begin{aligned} &\underset{\boldsymbol{\alpha}\in\mathbb{R}^n}{\text{maximize}} \quad -\frac{1}{2}\boldsymbol{\alpha}^T\mathbf{K}\boldsymbol{\alpha}+\boldsymbol{\alpha}^T\boldsymbol{y} \\ &\text{subject to} \quad \forall i=1\ldots n:\ \alpha_j\cdot y_j\geq 0 \ \text{ and } \ \sum_{i=1}^{n}\alpha_i=0\ . \end{aligned} \tag{2.62}$$

Deriving the dual (kernelized) version of the optimization problem (2.52) can be done analogously leading to additional upper bound constraints: $0 \leq \alpha_i \cdot y_i \leq C$ (Bishop, 2006, p. 333). An insight into the influence of the parameter C can be obtained by utilizing Theorem A.2, which can be found in the appendix. The theorem offers an upper bound for the standard deviation of the soft decision function f in terms of the coefficients $\boldsymbol{\alpha}$. Applying the additional constraints $0 \leq \alpha_i \cdot y_i \leq C$ that appear in the soft margin version of the dual problem (2.62) we can rewrite the bound as follows:

$$\sigma_{\boldsymbol{x}}\left(f(\boldsymbol{x})\right) \leq ||\boldsymbol{\alpha}|| \cdot \sqrt{\lambda_{\max}(\mathbf{K})}\cdot\zeta_K \leq \sqrt{n}\cdot C\cdot\sqrt{\lambda_{\max}(\mathbf{K})}\cdot\zeta_K \ , \tag{2.63}$$

with ζ_K being a kernel-dependent term and $\lambda_{\max}$ denoting the largest eigenvalue. This bound illustrates the influence of C on the decision function. Decreasing the parameter leads to a small deviation of f and thus to classification models of low complexity, such as hyperplanes in the original space. A high value of C offers a large variability of the decision function shape.

An important aspect when comparing the primal optimization problem (2.48) with its dual counterpart (2.62) is the change of the problem size from the input vector dimension D and an additional bias to the number of training examples n. Note that the effective size of the dual problem is the number of support vectors (Schölkopf and Smola, 2001) and due to this reason SVM is sometimes referred to as semi-parametric. Solving the dual instead of the primal problem is beneficial even for linear kernels when dealing with a small amount of training data and a large set of features. However, for large-scale learning scenarios, the direct usage of the kernel trick is impractical. Due to this reason, Vempati et al. (2010) and Li et al. (2010) propose to calculate an approximate feature transformation ϕ of a given kernel, which is used to apply a linear SVM classifier on explicitly transformed training examples.

After estimating the coefficients α_i by solving the dual problem (2.62), classifying a new example $\boldsymbol{x}_*$ consists of evaluating the hyperplane function

$f(\boldsymbol{x}_*)$:

$$f(\boldsymbol{x}_*) = \langle \boldsymbol{w}, \phi(\boldsymbol{x}_*) \rangle_{\mathcal{H}} + b = \sum_{i=1}^{n} \alpha_i K(\boldsymbol{x}_i, \boldsymbol{x}_*) + b \ , \tag{2.64}$$

and thresholding it at zero:

$$h(\boldsymbol{x}_*) = \text{sign}\left(f(\boldsymbol{x}_*)\right) = \text{sign}\left(\sum_{i=1}^{n} \alpha_i K(\boldsymbol{x}_i, \boldsymbol{x}_*) + b\right) \ . \tag{2.65}$$

The bias term b can also be expressed in terms of kernel evaluations, but we skip the derivation and refer to Schölkopf and Smola (2001) and Bishop (2006). Instead of a hard classification decision, a soft decision of the classifier is often necessary, such as in applications that use classification outputs as intermediate steps or for the multi-class classification approaches presented in the next section. For SVM approaches the continuous value $f(\boldsymbol{x}_*)$ can be used directly as a score value that relates to the likelihood of $\boldsymbol{x}_*$ being a positive example.

2.5.4 Multi-Class Classification

Up to this point, we considered binary classification tasks with $y_i \in \{-1, 1\}$. Learning a SVM classifier for a multi-class classification problem with an extended label space $y_i \in \{1, \ldots, M\}$ can be done in multiple ways. In general, we distinguish between approaches that try to formulate an optimization problem similar to the binary case, and those that build upon several binary classifiers trained on derived sub-problems. For the first category of methods, we refer to Schölkopf and Smola (2001, Section 7.6.4) and restrict the following description to the latter type.

Let a training dataset $\mathcal{D} = \{(\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_n, y_n)\}$ with $y_i \in \{1, \ldots, M\}$ and $M > 2$ be given. The *one-vs-all* or *one-vs-rest* approach trains M binary SVM classifiers f^j that try to separate each class j from all other classes. Thus, classifiers are taught using binary training sets $\mathcal{D}^j = \{(\boldsymbol{x}_1, \tilde{y}_1), \ldots, (\boldsymbol{x}_n, \tilde{y}_n)\}$ with $\tilde{y}_i = 1$ if $y_i = j$ and $\tilde{y}_i = -1$ otherwise. Classification of a new example $\boldsymbol{x}_*$ is done by evaluating every binary classifier and choosing the one with maximum score $f^j(\boldsymbol{x}_*)$. Therefore, the final decision of the multi-class classifier h can be written as:

$$h(\boldsymbol{x}_*) = \underset{j=1\ldots M}{\text{argmax}} \, f^j(\boldsymbol{x}_*) \ . \tag{2.66}$$

An alternative formulation applies the algorithm of Platt (1999) beforehand and chooses the category j that maximizes the approximated posterior probability.

Another popular idea is the *one-vs-one* method. As the name already suggests, $\binom{M}{2}$ classifiers $f^{j,j'}$ are learned, for each of the binary sub-problems that separate two classes j and j'. During testing, all classifiers are evaluated and for each class j we count the number of times z_j a binary classifier $f^{j,j'}$ classifies the new example $\boldsymbol{x}_*$ as being from j. A category that achieves the maximum count can be considered as the final output of the multi-class classifier. In contrast to the one-vs-all approach, deriving probability estimates is straightforward by normalizing the counts z_j. In our experiments, we use the one-vs-all method due to its advantages with respect to the computation time.

2.5.5 Hyperparameter optimization

The problem of hyperparameter optimization was already mentioned in Section 2.4.2 in the context of kernel parameters. For the SVM based methods presented in the previous sections, we additionally introduced the parameter $C > 0$ which weighted the penalty term corresponding to the slack variables $\boldsymbol{\xi}$.

The common method to optimize C and hyperparameters $\boldsymbol{\eta}$ of the kernel function, such as the parameter γ of the Gaussian kernel (*cf.* Eq. (2.41)), is k-fold cross-validation. To apply this technique, the training set $\mathcal{D}$ is divided into k subsets $\mathcal{D}^j$ and learning and testing is performed k times. In each round j, dataset $\mathcal{D}^j$ is used for testing the classifier learned on all remaining examples $\mathcal{D} \setminus \mathcal{D}^j$. After this procedure, we have k values of a performance measure, which are averaged to yield a final performance value of current parameter values. Searching optimal hyperparameters can be done by greedy optimization techniques that simply evaluate cross-validation performance on a predefined grid. Such a procedure is computationally demanding and impractical for more than two hyperparameters. Alternative methods are multiple kernel learning approaches (Sonnenburg et al., 2006) or hyperparameter optimization with Gaussian processes as described in Section 2.6.10.

2.6 Machine Learning with Gaussian Processes

The following section explains the usage of Gaussian processes (GP) for machine learning tasks, like regression and classification. The textbook of Rasmussen and

Williams (2005) is one of the key references for learning with GP models. There are two main possible derivations leading to the GP framework: the weight space view and the process view[3]. In the following presentation, we prefer the process view.

2.6.1 Gaussian Processes

A Gaussian process is roughly speaking a generalization of a multivariate normal distribution to infinite dimensions. It belongs to the much wider class of stochastic processes, which are collections of random variables. Defining a Gaussian process is done by considering all finite subsets of the input space, similar to the definition of positive definite kernels.

Definition 2.5 (Gaussian process) *A collection of random variables* $f = (f(\boldsymbol{x}))_{\boldsymbol{x}\in\mathcal{X}}$ *is a* Gaussian process *with mean function* $\mu : \mathcal{X} \to \mathbb{R}$ *and covariance function* $K : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$*, if and only if for every finite set* $\mathbf{X} = \{\boldsymbol{x}_i\}_{i=1}^n \subseteq \mathcal{X}$ *the corresponding random variables are jointly normally distributed, i.e.*

$$\boldsymbol{f} = (f(\boldsymbol{x}_i))_{i=1}^n \sim \mathcal{N}(\boldsymbol{\mu}, \mathbf{S})$$

with $\boldsymbol{\mu} = (\mu(\boldsymbol{x}_1), \ldots, \mu(\boldsymbol{x}_n))^T$ *and covariances* $S_{ij} = \mathbb{E}(f(\boldsymbol{x}_i)f(\boldsymbol{x}_j)) = K(\boldsymbol{x}_i, \boldsymbol{x}_j)$*. We use the notation* $f \sim \mathcal{GP}(\mu, K)$ *to refer to this definition.*

In the following, we restrict the presentation to zero-mean GPs, *i.e.* $\mu \equiv 0$, without loss of generality. There is a tight connection between covariance and kernel functions as introduced in Section 2.4. A valid covariance function has to induce valid covariance matrices of finite subsets, thus, it requires their positive definiteness. For this reason, each positive definite kernel function as defined by definition 2.1 is a valid covariance function of a Gaussian process and vice versa. In the remaining part of this thesis, we use both terms as synonyms.

The important advantage of Gaussian processes compared to a large set of other well-known stochastic processes, such as Poisson or gamma processes, is that they are defined even for multi-dimensional input sets $\mathcal{X}$ instead of being restricted to one-dimensional time domains. This offers the possibility to use a Gaussian process as a probability distribution over functions. The kernel function K of a GP determines the covariance of two function values $f(\boldsymbol{x})$ and $f(\boldsymbol{x}')$.

[3] Rasmussen and Williams (2005) refers to the process view as function-space view, but we follow Seeger (2003) by using the term "process view".

2.6.2 Sample Paths and the Influence of Hyperparameters

Figure 2.5 shows one-dimensional sample functions (sample paths) of zero-mean Gaussian processes with different kernel functions and varying hyperparameters. We also included a lower and upper function that bound a shaded area derived from three times the standard deviation of the corresponding GP. First of all, let us have a look at the first row, which resulted from using a GP with a Gaussian kernel function as defined by Eq. (2.41). Due to the stationarity of this kernel, the behavior of the sampled functions is independent of the absolute location. We also observe that the local variability of the function increases with an increasing value of the hyperparameter γ. This behavior be analyzed by considering an approximation of the expected quadratic gradient of f. Let us consider a set of points $x_1 < \ldots < x_n \in \mathbb{R}$ and the corresponding multivariate Gaussian random variable $\boldsymbol{f} = (f(x_1), \ldots, f(x_n))^T \in \mathbb{R}^n$. This allows us to reduce the expectation with respect to all possible functions f, to a well-defined expectation with respect to all n-dimensional vectors $\boldsymbol{f}$ and we avoid dealing with scary infinite-dimensional integrals. We can now study the following expected quadratic difference quotient:

$$g_K \stackrel{\text{def}}{=} \int_{\mathbb{R}^n} \left(\frac{1}{n-1} \sum_{i=1}^{n-1} \left(\frac{f(x_{i+1}) - f(x_i)}{x_{i+1} - x_i} \right)^2 \right) \cdot p(\boldsymbol{f})\, d\boldsymbol{f} \tag{2.67}$$

$$= \frac{1}{n-1} \sum_{i=1}^{n-1} \left(\frac{1}{(x_{i+1} - x_i)^2} \int_{\mathbb{R}^n} \left(f(x_{i+1}) - f(x_i)\right)^2 p(\boldsymbol{f})\, d\boldsymbol{f} \right) \tag{2.68}$$

$$= \frac{1}{n-1} \sum_{i=1}^{n-1} \frac{K(x_{i+1}, x_{i+1}) - 2K(x_{i+1}, x_i) + K(x_i, x_i)}{(x_{i+1} - x_i)^2} \ . \tag{2.69}$$

We now consider the Gaussian kernel and assume equidistant x_i with spacing h:

$$g_{\text{gauss}}(h) \stackrel{\text{def}}{=} g_K(x_1, x_1 + h, x_1 + 2h, \ldots) = \frac{2}{h^2} \left(1 - \exp\left(-\gamma h^2\right)\right) \ . \tag{2.70}$$

By using the rule of l'Hopital, we have in the limit:

$$g_{\text{gauss}} \stackrel{\text{def}}{=} \lim_{h \to 0} g_{\text{gauss}}(h) = \lim_{h \to 0} \frac{4\gamma h \exp\left(-\gamma h^2\right)}{2h} = 2\gamma \ . \tag{2.71}$$

Thereby, we have verified our intuition obtained from the first row of Figure 2.5 that local variations are intensified with an increasing value of the hyperparameter

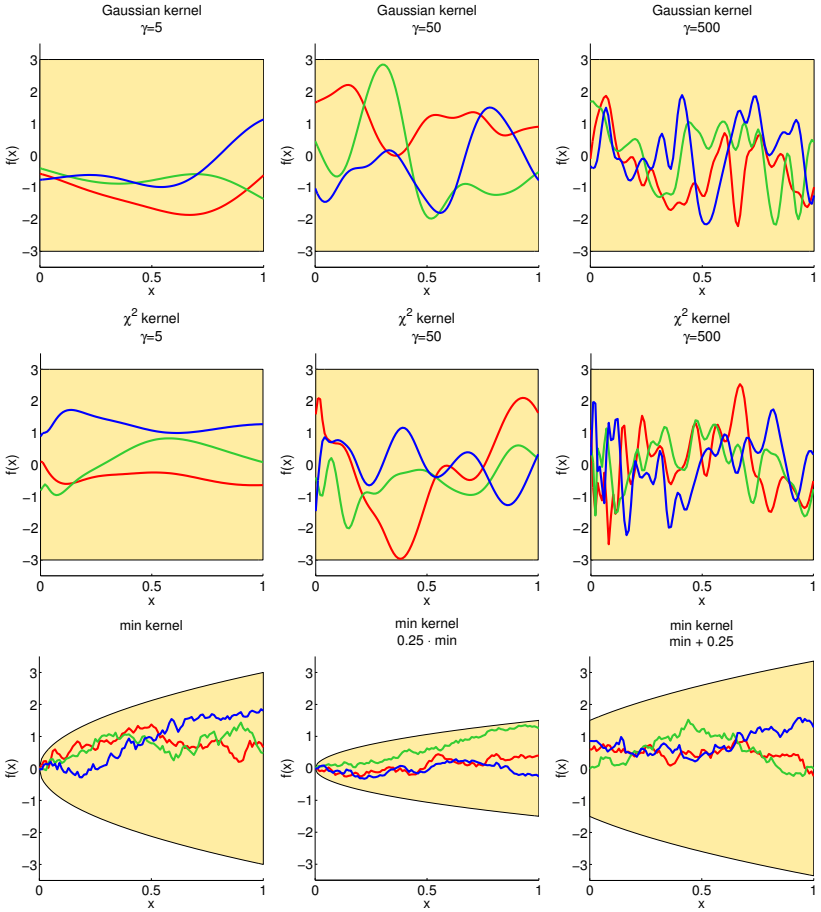


Figure 2.5: Three sample functions from zero-mean Gaussian processes with different kernels (rows: Gaussian kernel, χ^2-kernel, minimum intersection kernel) and varying kernel hyperparameters (columns). The shaded area highlights the interval derived from three times the standard deviation of the Gaussian process.

γ. The second row of Figure 2.5 shows sample functions obtained using the χ^2-kernel (Eq. (2.42)). The behavior is very similar to the Gaussian kernel except that the non-stationarity of this kernel can be observed slightly. Local variations of the function are much more common in the first part of the plot next to the origin.

Characteristic sample paths of a GP equipped with a minimum intersection kernel (Eq. (2.43)) are displayed on the bottom row of Figure 2.5. For one-dimensional input spaces $\mathcal{X} = \mathbb{R}$, a GP with a minimum intersection kernel is equivalent to the famous Wiener process. The interesting property of this process is that sampled functions are continuous in the mean-square sense but not differentiable (Seeger, 2003, Section 2.1.1.3). One effect of this property can be seen by considering our measure $g_K(x_1, \ldots, x_n)$:

$$g_{\min}(x_1, \ldots, x_n) = \frac{1}{n-1} \sum_{i=1}^{n-1} \frac{x_{i+1} - 2x_i + x_i}{(x_{i+1} - x_i)^2} = \frac{1}{n-1} \sum_{i=1}^{n-1} \frac{1}{x_{i+1} - x_i} \quad . \tag{2.72}$$

Therefore, $g_{\min}$ does not have a finite limit for increasingly finer grids, which is obvious for equidistant grids with spacing h leading to $g_{\min}(x_1, \ldots, x_n) = \frac{1}{h}$. The standard deviation of the function value $f(x)$ is $\sqrt{x}$ and also leads to the property that $f(0) = 0$ holds almost surely. In contrast to the Gaussian or the χ^2-kernel, the minimum intersection kernel does not have an internal parameter γ. Therefore, the plots show the influence of the scale and bias parameter introduced in Eq. (2.36).

2.6.3 Basic Principle

As we have seen in Section 2.1, machine learning can be described as estimating the relation between inputs $\boldsymbol{x}$ and outputs y. This relation is often described using a real-valued function f, *e.g.* $y = f(\boldsymbol{x}) + \varepsilon$ with ε being a noise term. A large set of machine learning approaches, like support vector machines (Section 2.5) or decision trees (Section 2.3.1), parameterize the latent function f by $f(\cdot; \boldsymbol{\theta})$ and estimate the parameters $\boldsymbol{\theta}$, *e.g.* with maximum likelihood or maximum a posteriori estimation (Section 2.2.1).

In contrast, the following machine learning approaches follow the idea of marginalization as introduced in Section 2.2.2. An important ingredient of marginalization is a model of suitable prior distributions $p(\boldsymbol{\theta})$. Instead of parameterizing f and marginalizing $\boldsymbol{\theta}$, Gaussian process models allow marginalizing

the latent function f directly without any parameterization. This is due to their ability to model a large variety of prior distributions $p(f)$ of real-valued functions using kernel functions. Especially Gaussian processes resulting in continuous and smooth sample paths are of interest, because they offer the possibility to incorporate one of the fundamental assumptions in machine learning: similar inputs should lead to similar outputs.

2.6.4 Model Assumptions

First of all, let us have a look at the two main assumptions of Gaussian processes as used for regression and classification:

1. **Conditional Independence:** There is an underlying latent function $f : \mathcal{X} \to \mathbb{R}$, such that outputs y are conditionally independent from the input $\boldsymbol{x}$ given the latent function value $f(\boldsymbol{x})$. The outputs are distributed according to the so called *noise model* $p(y \mid f(\boldsymbol{x}))$.
2. **Prior Model:** The function f is a sample of a GP prior and is in the following represented as a random variable $f \sim \mathcal{GP}(\mathbf{0}, K)$ with zero mean and kernel function $K : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$.

Let n training examples $\boldsymbol{x}_i \in \mathbf{X} \subset \mathcal{X}$ be given with outputs or labels $y_i \in \mathcal{Y}$ collected in a vector $\boldsymbol{y} \in \{-1, 1\}^n$. We would like to predict the posterior of the output y_* of an unseen example $\boldsymbol{x}_* \in \mathcal{X}$ by:

$$p(y_* \mid \boldsymbol{x}_*, \boldsymbol{y}, \mathbf{X}) = \int_{\mathbb{R}} p(y_* \mid f_*)\; p(f_* \mid \boldsymbol{x}_*, \boldsymbol{y}, \mathbf{X})\; df_* \;\;, \tag{2.73}$$

where we marginalized the latent function value $f_* = f(\boldsymbol{x}_*)$ corresponding to $\boldsymbol{x}_*$. Note that we made use of the conditional independence assumption. The conditional distribution of f_* is also available by marginalizing all latent function values $\boldsymbol{f} = (f(\boldsymbol{x}_i))_{i=1}^n$ of the training set $\mathbf{X}$:

$$p(f_* \mid \boldsymbol{x}_*, \boldsymbol{y}, \mathbf{X}) = \int_{\mathbb{R}^n} p(f_* \mid \boldsymbol{x}_*, \boldsymbol{f})\; p(\boldsymbol{f} \mid \boldsymbol{y}, \mathbf{X})\; d\boldsymbol{f} \;\;. \tag{2.74}$$

Finally, the incorporation of the noise model and our assumption of independent outputs leads to:

$$p(\boldsymbol{f} \mid \boldsymbol{y}, \mathbf{X}) = \frac{p(\boldsymbol{f} \mid \mathbf{X})}{p(\boldsymbol{y} \mid \mathbf{X})} \left(\prod_{i=1}^{n} p(y_i \mid f_i) \right) \;\;. \tag{2.75}$$

The distribution $p(\boldsymbol{f} \mid \mathbf{X})$ is a n-dimensional normal distribution with zero mean and covariance matrix $\mathbf{K} = (K(\boldsymbol{x}_i, \boldsymbol{x}_j))_{i,j}$. The noise model $p(y_i \mid f_i)$ can take various forms depending on the nature of the outputs y_i, which leads to the distinction of Gaussian process regression and classification.

2.6.5 GP Regression

In the following, we show how to utilize the GP framework to solve regression tasks, *i.e.* with real-valued outputs $y \in \mathbb{R}$. A suitable and common noise model for regression tasks is additive zero-mean Gaussian noise. Error terms ε for each input $\boldsymbol{x}$ are assumed to be independent and identically distributed according to $\varepsilon \sim \mathcal{N}(0, \sigma_\varepsilon^2)$, which leads to:

$$p(y \mid f(\boldsymbol{x})) = \mathcal{N}(y \mid f(\boldsymbol{x}), \sigma_\varepsilon^2) \ . \tag{2.76}$$

The assumption of identically distributed noise is often referred to as *homogeneous noise*, whereas noise terms depending on inputs are called *heteroscedastic*. Due to the Gaussian noise assumption, we can derive a closed form inference equation for Gaussian processes. This important property results from the observation that outputs y are the sum of two Gaussian random variables $f(\boldsymbol{x})$ and ε and are therefore also Gaussian themselves. For this reason, the joint distribution of the output y_* of a new example $\boldsymbol{x}_*$ and training outputs $\boldsymbol{y}$ can be given as follows:

$$\begin{aligned} p\left(y_*, \boldsymbol{y} \mid \mathbf{X}\right) &= \mathcal{N}\left(\begin{bmatrix} y_* \\ \boldsymbol{y} \end{bmatrix} \mid \mathbf{0}, K\left((\boldsymbol{x}_*, \mathbf{X}), (\boldsymbol{x}_*, \mathbf{X})\right) + \sigma_\varepsilon^2 \cdot \mathbf{I} \right) \\ &= \mathcal{N}\left(\begin{bmatrix} y_* \\ \boldsymbol{y} \end{bmatrix} \mid \mathbf{0}, \begin{bmatrix} K(\boldsymbol{x}_*, \boldsymbol{x}_*) + \sigma_\varepsilon^2 & \boldsymbol{k}_*^T \\ \boldsymbol{k}_* & \mathbf{K} + \sigma_\varepsilon^2 \cdot \mathbf{I} \end{bmatrix} \right) \ , \end{aligned}$$

with zero mean vector and covariance matrix computed by the kernel function K and an additional noise term. Using the derivations in Section A.4, we see that the posterior of y_* conditioned on $\boldsymbol{y}$ is also Gaussian, *i.e.* $y_* \sim \mathcal{N}(\mu_*, \sigma_*^2)$ with mean value:

$$\mu_* = \boldsymbol{k}_*^T \left(\mathbf{K} + \sigma_\varepsilon^2 \cdot \mathbf{I}\right)^{-1} \boldsymbol{y} = \boldsymbol{k}_*^T \boldsymbol{\alpha} \ , \tag{2.77}$$

and coefficients $\boldsymbol{\alpha} = \left(\mathbf{K} + \sigma_\varepsilon^2 \cdot \mathbf{I}\right)^{-1} \boldsymbol{y}$. The mean value μ_* is also the mode of the posterior and thus used as the predicted value for a given input vector

$\boldsymbol{x}_*$. We refer to it as the predictive mean or regression function. An important benefit of the GP framework, due to its utilization of exact Bayesian inference (Section 2.2.2), is the availability of the predictive variance, which is often used as a confidence or uncertainty estimate[4]:

$$\sigma_*^2 = K(\boldsymbol{x}_*, \boldsymbol{x}_*) - \boldsymbol{k}_*^T \left(\mathbf{K} + \sigma_\varepsilon^2 \cdot \mathbf{I}\right)^{-1} \boldsymbol{k}_* + \sigma_\varepsilon^2 \ . \tag{2.78}$$

Calculating the predictive mean involves computing the coefficients $\boldsymbol{\alpha}$ in advance, *e.g.* by using the Cholesky factorization of the regularized kernel matrix requiring $\mathcal{O}(n^3)$ operations, and computing the scalar product $\boldsymbol{k}_*^T \boldsymbol{\alpha}$ for each test input $\boldsymbol{x}_*$ in linear time. The posterior variance needs $\mathcal{O}(n^2)$ for each new input using the pre-computed Cholesky factor.

Figure 2.6 shows an example of GP regression applied to a one-dimensional toy example utilizing the Gaussian kernel and three different values of the hyperparameter γ. First of all, let us have a look at the predictive mean function, which is displayed as a red graph in all of the plots. In the top plot, we can see the effect of the noise model. The regression function does not have to include all given training points, because we assumed that they are corrupted by additive noise $\varepsilon \sim \mathcal{N}(0, \sigma_\varepsilon^2)$. The shape of the function strongly depends on the used hyperparameter γ and directly reflects our analysis of the expected quadratic gradient of functions sampled from the GP prior (Section 2.6.1). With an increasing value of γ the GP prior allows a higher flexibility of the predictive mean function. The shaded area in the plots corresponds to the confidence area $[-3\sigma_*, 3\sigma_*]$ calculated using the posterior variance σ_*^2. A high variance of the posterior at a point $\boldsymbol{x}_*$ is observed when $\boldsymbol{x}_*$ is distant from the training points, *e.g.* in the border areas of the plot. Due to the positive definiteness of $\left(\mathbf{K} + \sigma_\varepsilon^2 \cdot \mathbf{I}\right)$ and the uniformity property $K(\boldsymbol{x}, \boldsymbol{x}) = 1$ of the Gaussian kernel, $1 + \sigma_\varepsilon^2$ is an upper bound of the posterior variance and also the limit value as the minimum distance to the training points approaches infinity.

2.6.6 GP Classification

We have seen that GP regression with a Gaussian noise model leads to closed-form solutions of the underlying marginalization in Eq. (2.73) and Eq. (2.74). For binary classification tasks with $y = \{-1, 1\}$, GP regression can be directly

[4] From an information theoretical perspective, the uncertainty is the entropy of the distribution. For Gaussian distributions with variance σ^2, the differential entropy is exactly $\frac{1}{2}\log(2\pi e\sigma^2)$ showing the strong relationship between the uncertainty and the variance.

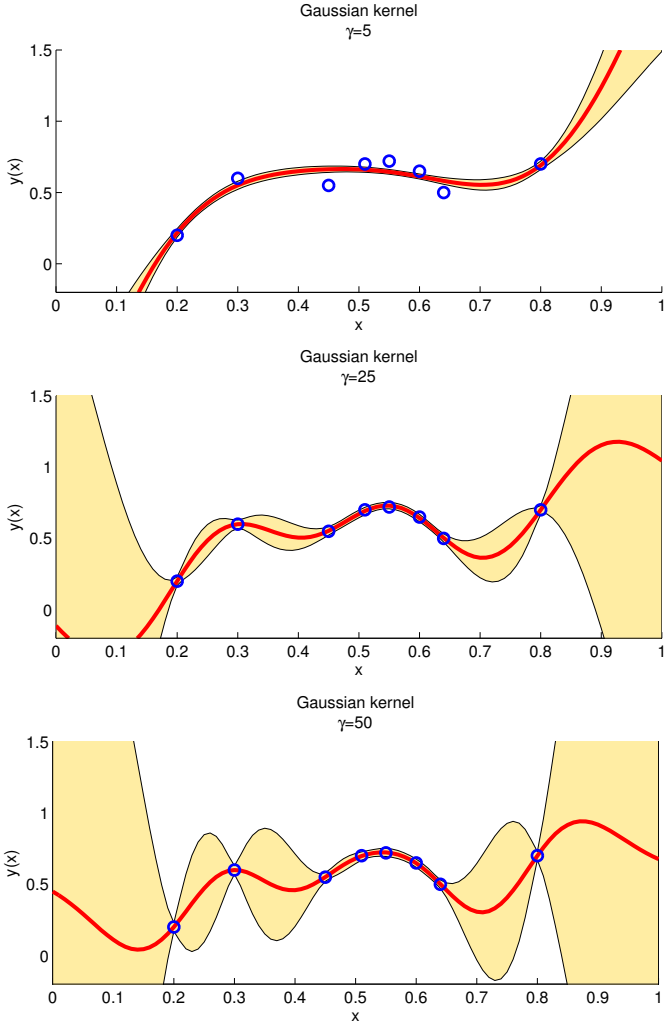


Figure 2.6: Examples of Gaussian process regression with the Gaussian kernel (2.41) and a varying kernel hyperparameter γ. The red graph displays the predictive mean function (2.77) and the shaded area highlights the interval derived from three times the posterior standard deviation. The standard deviation of the noise was set to 0.01.

applied as a classification technique and the predictive mean or the density value $p(y_* = 1 \,|\, \boldsymbol{x}_*, \boldsymbol{y}, \mathbf{X})$ serves as a smooth classification score, which is thresholded to do a final classification decision. This approach is called *label regression* (Nickisch and Rasmussen, 2008) and involves only a few basic algebraic operations. However, a Gaussian noise model seems not to be appropriate, because it completely neglects the discrete nature of y and treats it as a continuous random variable. In contrast, the noise model should obey:

$$\forall f \in \mathbb{R} \quad : \quad p(y = 1 \,|\, f) + p(y = -1 \,|\, f) = 1 \ . \tag{2.79}$$

One possibility is to use the *probit model* that is justified by a classification decision made by thresholding the latent function value, *i.e.* $y \stackrel{\text{def}}{=} \text{sign}\,(f + \varepsilon)$ with $\varepsilon \sim \mathcal{N}(0, \sigma_c^2)$. The resulting discrete distribution of y is expressed using the cumulative Gaussian function Φ, which is related to the error function $\text{erf}(z)$:

$$p(y \,|\, f) = \begin{cases} p(f + \epsilon \geq 0) & \text{if } y = 1 \\ p(f + \epsilon < 0) & \text{if } y = -1 \end{cases} \tag{2.80}$$

$$= \int_{-\infty}^{y \cdot f} \mathcal{N}(z \,|\, 0, \sigma_c^2)\, dz \tag{2.81}$$

$$= \frac{1}{2}\left(\text{erf}\left(\frac{y \cdot f}{\sqrt{2} \cdot \sigma_c}\right) + 1\right) = \Phi\left(\frac{y \cdot f}{\sigma_c}\right) \ . \tag{2.82}$$

In most cases the scaling factor σ_c of the noise model is set to one, because it can be substituted by additional multiplicative scaling of the kernel function. An alternative is to utilize the sigmoid function often referred to as the logistic noise model (Rasmussen and Williams, 2005):

$$p(y \,|\, f) \stackrel{\text{def}}{=} \frac{1}{2}\left(\text{sig}\,(y \cdot f) + 1\right) \ . \tag{2.83}$$

The logistic model is widely used for neural networks (Bishop, 2006, Section 5).

The disadvantage of both classification models is that the conditional distribution $p(\boldsymbol{f} \,|\, \boldsymbol{y}, \mathbf{X})$ (Eq. (2.75)) of latent function values $\boldsymbol{f}$ is not Gaussian anymore and the marginalization (2.74) becomes intractable to compute. This problem is a typical one for exact Bayesian marginalization and was already discussed in Section 2.2.2. In the following sections, we briefly review Laplace approximation as an important approximate inference technique that is used to solve the involved marginalizations for non-Gaussian noise models. Another

important method for approximate inference is expectation propagation (Minka, 2001) and we refer the reader to (Rasmussen and Williams, 2005, Section 3.6) for a description of its application to GP classification.

2.6.7 Laplace Approximation

We focus on handling the marginalization of the multivariate random variable $\boldsymbol{f}$. The one-dimensional integration needed to marginalize f_* (Eq. (2.73)) can be done with simple Monte Carlo techniques and even has a closed-form solution in the case of the probit model (Rasmussen and Williams, 2005, Section 3.4.2).

If the conditional distribution $p(\boldsymbol{f} \mid \boldsymbol{y}, \mathbf{X})$ is Gaussian, marginalizing $\boldsymbol{f}$ in Eq. (2.74) can be done in a computationally efficient manner. The main idea of the method of Laplace is to approximate a non-Gaussian distribution with a Gaussian distribution $q(\boldsymbol{f} \mid \boldsymbol{y}, \mathbf{X})$. If q is Gaussian, $\log q$ is a second order function. Therefore, to find q, we can use the Taylor approximation of $L(\boldsymbol{f}) \stackrel{\text{def}}{=} \log p(\boldsymbol{f} \mid \boldsymbol{y}, \mathbf{X})$. Let $\hat{\boldsymbol{f}}$ be the mode of the exact conditional distribution, *i.e.* the maximum a posteriori estimate, which can be found by a small number of Newton iterations (Rasmussen and Williams, 2005). The Taylor approximation of L around $\hat{\boldsymbol{f}}$ is as follows and can be simplified due to the zero gradient at $\hat{\boldsymbol{f}}$:

$$\begin{aligned} L(\boldsymbol{f}) &= L(\hat{\boldsymbol{f}}) + \left[\{\nabla L\}\left(\hat{\boldsymbol{f}}\right)\right]^T \left(\boldsymbol{f} - \hat{\boldsymbol{f}}\right) \\ &\quad + \frac{1}{2}\left(\boldsymbol{f} - \hat{\boldsymbol{f}}\right)^T \left[\{\nabla^2 L\}\left(\hat{\boldsymbol{f}}\right)\right]\left(\boldsymbol{f} - \hat{\boldsymbol{f}}\right) + \dots \qquad (2.84) \\ &= L(\hat{\boldsymbol{f}}) + \frac{1}{2}\left(\boldsymbol{f} - \hat{\boldsymbol{f}}\right)^T \left[\{\nabla^2 L\}\left(\hat{\boldsymbol{f}}\right)\right]\left(\boldsymbol{f} - \hat{\boldsymbol{f}}\right) + \dots \ , \qquad (2.85) \end{aligned}$$

which leads to the following approximation of the conditional distribution:

$$q(\boldsymbol{f} \mid \boldsymbol{y}, \mathbf{X}) = \mathcal{N}\left(\boldsymbol{f} \mid \hat{\boldsymbol{f}}, -\left[\{\nabla^2 L\}\left(\hat{\boldsymbol{f}}\right)\right]^{-1}\right) \ . \qquad (2.86)$$

The positive definiteness of the covariance matrix is guaranteed because of $\hat{\boldsymbol{f}}$ being a local maximum. Let us derive the exact formula of the Hesse matrix using the definition of L and Bayes' law:

$$\begin{aligned} L(\boldsymbol{f}) &= \log p(\boldsymbol{y} \mid \boldsymbol{f}) + \log p(\boldsymbol{f} \mid \mathbf{X}) - \log p(\boldsymbol{y} \mid \mathbf{X}) \qquad (2.87) \\ \left\{\nabla^2 L\right\}\left(\hat{\boldsymbol{f}}\right) &= \left\{\nabla^2_{\boldsymbol{f}} \log p(\boldsymbol{y} \mid \boldsymbol{f})\right\}(\hat{\boldsymbol{f}}) - \mathbf{K}^{-1} \qquad (2.88) \\ &= -\mathbf{W} - \mathbf{K}^{-1} \ , \qquad (2.89) \end{aligned}$$

with $\mathbf{W}$ being the $n \times n$ diagonal matrix containing the second derivatives of the negative logarithm of the noise model:

$$\forall 1 \leq i \leq n \quad : \quad W_{ii} = -\left\{ \frac{d^2}{d^2 f} \log p(y_i \mid f) \right\} (\hat{f}_i). \tag{2.90}$$

The matrix is diagonal due to the assumption that labels are conditionally independent given the corresponding function value. For inference, we need to derive the posterior distribution of the latent function value f_*, but first of all let us summarize our knowledge about the parts of the integrand in Eq. (2.74):

$$p(f_* \mid \boldsymbol{x}_*, \boldsymbol{f}) = \mathcal{N}\left(f_* \mid \boldsymbol{k}_*^T \mathbf{K}^{-1} \boldsymbol{f},\ K(\boldsymbol{x}_*, \boldsymbol{x}_*) - \boldsymbol{k}_*^T \mathbf{K}^{-1} \boldsymbol{k}_*\right) \quad , \tag{2.91}$$

$$p(\boldsymbol{f} \mid \boldsymbol{y}, \mathbf{X}) \approx q(\boldsymbol{f} \mid \boldsymbol{y}, \mathbf{X}) = \mathcal{N}\left(\boldsymbol{f} \mid \hat{\boldsymbol{f}}, \left(\mathbf{W} + \mathbf{K}^{-1}\right)^{-1}\right) \quad . \tag{2.92}$$

We are now ready to apply Lemma A.7, which gives us the approximated predictive mean of f_*:

$$\mathbb{E}\left(f_* \mid \boldsymbol{x}_*, \boldsymbol{y}, \mathbf{X}\right) = \boldsymbol{k}_*^T \mathbf{K}^{-1} \hat{\boldsymbol{f}} \quad , \tag{2.93}$$

and the corresponding posterior variance:

$$\begin{aligned} \sigma^2\left(f_* \mid \cdot\right) &= K(\boldsymbol{x}_*, \boldsymbol{x}_*) - \boldsymbol{k}_*^T \mathbf{K}^{-1} \boldsymbol{k}_* + \boldsymbol{k}_*^T \mathbf{K}^{-1} \left(\mathbf{W} + \mathbf{K}^{-1}\right)^{-1} \mathbf{K}^{-1} \boldsymbol{k}_* \\ &= K(\boldsymbol{x}_*, \boldsymbol{x}_*) - \boldsymbol{k}_*^T \left(\mathbf{K}^{-1} - \mathbf{K}^{-1} \left(\mathbf{W} + \mathbf{K}^{-1}\right)^{-1} \mathbf{K}^{-1}\right) \boldsymbol{k}_* \\ \text{(Lemma A.4)} &= K(\boldsymbol{x}_*, \boldsymbol{x}_*) - \boldsymbol{k}_*^T \left(\mathbf{K} + \mathbf{W}^{-1}\right)^{-1} \boldsymbol{k}_* \quad . \end{aligned} \tag{2.94}$$

For implementation details, especially to improve the numerical stability, we refer to Rasmussen and Williams (2005, Section 3.4.3).

2.6.8 Relationship to Other Methods

In the following, we briefly review similarities and differences between classification with GP prior models and other methods. The presented results are important for the later presentation in Section 3.4.6 of connections between different transfer learning methods.

2.6.8.1 Relationship to SVM Classification

Classification with GP models is strongly connected to the kernel version of SVM presented in Section 2.5. To see the exact similarities, we first rewrite the SVM optimization problem (2.54) in its kernel version using Eq. (2.60) and follow Rasmussen and Williams (2005, p. 144) by assuming that the bias b was incorporated in the kernel function:

$$\underset{\boldsymbol{\alpha}\in\mathbb{R}^n}{\text{minimize}} \quad \sum_{i=1}^{n} H\left(y_i \cdot f(\boldsymbol{x}_i)\right) + \lambda\, \boldsymbol{\alpha}^T \mathbf{K} \boldsymbol{\alpha} \ . \tag{2.95}$$

The GP framework is formulated in terms of the latent function f, which can also be done for the SVM optimization problem (2.95) using $\mathbf{K}\boldsymbol{\alpha} = \boldsymbol{f}$ derived from Eq. (2.64):

$$\underset{\boldsymbol{f}\in\mathbb{R}^n}{\text{minimize}} \quad C \sum_{i=1}^{n} H\left(y_i \cdot f_i\right) + \frac{1}{2}\, \boldsymbol{f}^T \mathbf{K}^{-1} \boldsymbol{f} \ . \tag{2.96}$$

We additionally used the fact that $\lambda = (2C)^{-1}$. A similar optimization problem arises in the GP framework when maximizing $p(\boldsymbol{f} \mid \boldsymbol{y}, \mathbf{X})$, *i.e.* minimizing $-\log p(\boldsymbol{f} \mid \boldsymbol{y}, \mathbf{X})$, with respect to $\boldsymbol{f}$:

$$\underset{\boldsymbol{f}\in\mathbb{R}^n}{\text{minimize}} \quad -\sum_{i=1}^{n} \log p(y_i \mid f_i) + \frac{1}{2} \boldsymbol{f}^T \mathbf{K}^{-1} \boldsymbol{f} \ . \tag{2.97}$$

Thus, the underlying objective functions are very similar and differ in the error terms depending on the product $z = y_i f_i$. SVM utilizes the hinge loss $H(z)$ without having a directly corresponding probabilistic model. GP regression combined with a Gaussian noise model uses the quadratic loss function $Q(z) = (1 - z)^2$ with $z = y_i f_i$. GP classification relies on $L_{\text{sig}}(z) = -\log \text{sig}(z)$ or $L_{\Phi}(z) = -\log \Phi(z)$. Figure 2.7 plots all of the mentioned loss functions and compares them to the misclassification error, which is also plotted in black color. We see that all loss functions bound the misclassification error from above with a continuous and smooth function.

If we use GP regression with Gaussian noise $p(y_i \mid f_i) = \mathcal{N}(y_i \mid f_i, \sigma_\varepsilon^2)$, we can directly rewrite (2.97) as:

$$\underset{\boldsymbol{f}\in\mathbb{R}^n}{\text{minimize}} \quad \frac{1}{2\sigma_\varepsilon^2} ||\boldsymbol{y} - \boldsymbol{f}||^2 + \frac{1}{2} \boldsymbol{f}^T \mathbf{K}^{-1} \boldsymbol{f} \ . \tag{2.98}$$

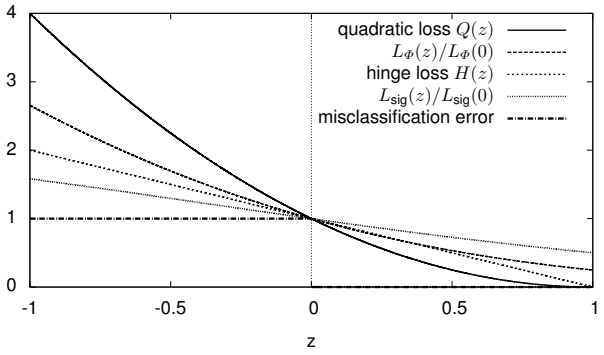


Figure 2.7: Visualization of different loss functions: hinge loss used by SVM, quadratic loss used by GP regression, L_Φ and L_{sig} utilized for GP classification. These loss functions are compared to the misclassification error function shown in black color. This figure is best viewed in color.

By comparing Eq. (2.98) and Eq. (2.96), it is apparent that C^{-1} and σ_ε^2 seem to have a similar effect on the objective function. Increasing the noise standard deviation σ_ε^2 gives more weight on the regularization term, thus, favors low complexity functions. This connection between C^{-1} and σ_ε^2 can be made more precise by applying Theorem A.2 using $\boldsymbol{\alpha} = \left(\mathbf{K} + \sigma_\varepsilon^2 \cdot \mathbf{I}\right)^{-1} \boldsymbol{y}$ and the fact that the spectral matrix norm[5] is induced by the Euclidean norm:

$$\sigma_{\boldsymbol{x}}\left(f(\boldsymbol{x})\right) \leq ||\boldsymbol{\alpha}|| \cdot \sqrt{\lambda_{\max}(\mathbf{K})} \cdot \zeta_K \tag{2.99}$$

$$= \left|\left|\left(\mathbf{K} + \sigma_\varepsilon^2 \cdot \mathbf{I}\right)^{-1} \boldsymbol{y}\right|\right| \sqrt{\lambda_{\max}(\mathbf{K})} \cdot \zeta_K \tag{2.100}$$

$$\leq \left|\left|\left(\mathbf{K} + \sigma_\varepsilon^2 \cdot \mathbf{I}\right)^{-1}\right|\right|_2 ||\boldsymbol{y}|| \sqrt{\lambda_{\max}(\mathbf{K})} \cdot \zeta_K \tag{2.101}$$

$$\leq \lambda_{\max}\left(\left(\mathbf{K} + \sigma_\varepsilon^2 \cdot \mathbf{I}\right)^{-1}\right) \cdot \sqrt{n} \cdot \sqrt{\lambda_{\max}(\mathbf{K})} \cdot \zeta_K \tag{2.102}$$

$$= \frac{\sqrt{\lambda_{\max}(\mathbf{K})}}{\lambda_{\min}(\mathbf{K}) + \sigma_\varepsilon^2} \cdot \sqrt{n} \cdot \zeta_K \quad . \tag{2.103}$$

[5]The spectral norm is defined as $||\mathbf{A}||_2 = \sqrt{\lambda_{\max}\left(\mathbf{A}^T\mathbf{A}\right)}$ which reduces for positive definite matrices to $\lambda_{\max}\left(\mathbf{A}\right)$.

Thus, similar to the weighting factor C, the resulting standard deviation of the underlying function f can be controlled by σ_ε^2.

2.6.8.2 Equivalence of Least-Squares SVM and GP Regression

Another variant of SVM is *least-squares SVM (LS-SVM)* (Suykens et al., 2002), which solves the following optimization problem to estimate the parameters of the linear classifier $f(\boldsymbol{x}) = \langle \boldsymbol{w}, \phi(\boldsymbol{x}) \rangle_{\mathcal{H}} + b$:

$$\begin{aligned} \underset{\boldsymbol{w} \in \mathbb{R}^D, b \in \mathbb{R}, \boldsymbol{\xi} \in \mathbb{R}^n}{\text{minimize}} \quad & \frac{1}{2} ||\boldsymbol{w}||_{\mathcal{H}}^2 + \frac{C}{2} \sum_{i=1}^{n} \xi_i^2 \\ \text{subject to} \quad & \forall i = 1 \ldots n: \; y_i = (\langle \boldsymbol{w}, \phi(\boldsymbol{x}_i) \rangle_{\mathcal{H}} + b) + \xi_i \quad . \end{aligned} \tag{2.104}$$

Note that this optimization problem is very similar to the soft margin optimization problem (2.52) of the SVM algorithm and differs in the quadratic use of the slack variables and modified constraints. The additional multiplication of the penalty parameter C with $\frac{1}{2}$ is simply due to mathematical convenience. In the following, we assume that the bias b is incorporated into the feature transformation ϕ and the kernel function K, *i.e.* we set $b = 0$. This is the same assumption as used in Section 2.6.8.1 and allows us to clarify the relationship. Incorporating the equality constraints leads to the following modified objective function:

$$L(\boldsymbol{w}) = \frac{1}{2} ||\boldsymbol{w}||^2 + \frac{C}{2} \sum_{i=1}^{n} \left(y_i - (\langle \boldsymbol{w}, \phi(\boldsymbol{x}_i) \rangle_{\mathcal{H}}) \right)^2 \quad . \tag{2.105}$$

The representer theorem shows that if we use the kernel trick and the above objective function, we can rewrite the hyperplane decision function with respect to the coefficients $\boldsymbol{\alpha} \in \mathbb{R}^n$:

$$f(\boldsymbol{x}_i) = \langle \boldsymbol{w}, \phi(\boldsymbol{x}_i) \rangle_{\mathcal{H}} = \sum_{j=1}^{n} \langle \alpha_j \phi(\boldsymbol{x}_j), \phi(\boldsymbol{x}_i) \rangle_{\mathcal{H}} = \sum_{j=1}^{n} \alpha_j K(\boldsymbol{x}_j, \boldsymbol{x}_i) = (\mathbf{K}\boldsymbol{\alpha})_i \quad . \tag{2.106}$$

We now use the abbreviation $\boldsymbol{f} \stackrel{\text{def}}{=} (f(\boldsymbol{x}_i))_{i=1}^n$ and the same arguments as used in Section 2.6.8, which lead after some simple manipulations to the following kernelized optimization problem

$$\underset{\boldsymbol{f} \in \mathbb{R}^D}{\text{minimize}} \quad \frac{C}{2} \cdot ||\boldsymbol{y} - \boldsymbol{f}||^2 + \frac{1}{2} \boldsymbol{f}^T \mathbf{K}^{-1} \boldsymbol{f} \quad . \tag{2.107}$$

By identifying C with $(\sigma_\varepsilon^2)^{-1}$, we can directly see the equivalence to the MAP formulation (2.98) of GP regression.

2.6.9 Multi-class Classification

So far, we presented GP classification with binary labels. However, some applications in this thesis require solving multi-class classification problems. Rasmussen and Williams (2005) present an extension of the Laplace Approximation for multi-class classification tasks with $y \in \mathcal{Y} = \{1, \ldots, M\}$. This method requires MCMC techniques and is thus computationally demanding.

Therefore, we follow Kapoor et al. (2010) by utilizing the one-vs-all technique, which has already been introduced for SVM in Section 2.5.4. For each class $k \in \mathcal{Y}$ a binary GP classifier is trained that uses all examples of k as positives and all remaining examples as a negative training set. Let $\boldsymbol{y}^k \in \{-1, 1\}^n$ be the vector of binary labels corresponding to class $k \in \{1, \ldots, M\}$ derived from the multi-class label vector $\boldsymbol{y}$ by $y_i^k = 1$ if $y_i = k$ and zero otherwise. For GP regression, the final predicted category is the one that achieves the highest predictive mean given by the corresponding binary problem:

$$y_*^{\text{multi}} = \underset{k=1\ldots M}{\operatorname{argmax}} \ \mu_*^k = \underset{k=1\ldots M}{\operatorname{argmax}} \ \boldsymbol{k}_*^T \left(\mathbf{K} + \sigma_\varepsilon^2 \cdot \mathbf{I}\right)^{-1} \boldsymbol{y}^k \ . \tag{2.108}$$

An important computational benefit is that binary classifiers are trained with different binary labels $\boldsymbol{y}$ but with the same set of input examples. For this reason, if we use the same hyperparameters for each classifier, the computed kernel matrix and its Cholesky factor can be utilized for every binary classifier. The Cholesky decomposition has to be done only once requiring the main computational effort of $\mathcal{O}(n^3)$ operations and the time needed for training a multi-class classifier is not dominated by the number of classes M.

2.6.9.1 Probability Calibration

In some applications, suitable probabilities for each class are important, because they can be easily used for further processing. The one-vs.-all approach only offers a hard classification decision as given in Eq. (2.108). To derive probability estimates for each class, we could squash the predictive means into some softmax function (Milgram et al., 2005). However, this strategy completely ignores the uncertainty of the estimate and hides the fact that the one-vs.-all decision is also probabilistic in its nature.

We propose to take the whole posterior distribution $\mathcal{N}(\mu_*^k, \sigma_*^2)$ of each random variable y_*^k into account. Thereby, the probability of class k achieving the maximum score can be expressed by:

$$p\left(y_*^{\text{multi}} = k\right) = p\left(\max_{k'=1\ldots M} y_*^{k'} = y_*^k\right) \quad . \tag{2.109}$$

Unfortunately, it seems not to be possible to derive a closed-form solution of the probability on the right hand side of Eq. (2.109) for a multi-class scenario with $M > 2$. Therefore, we use a simple Monte Carlo technique and sample Z times, *e.g.* $Z = 200$, from all M Gaussian distributions $\mathcal{N}(\mu_*^k, \sigma_*^2)$ and estimate the probability of each class k by $p(y_*^{\text{multi}} = k) \approx \frac{Z_k}{Z}$ with Z_k denoting the number of times the draw from y_*^k was the maximum value. A large variance σ_*^2, *i.e.* a high uncertainty of the estimate, leads to a nearly uniform distribution $p(y_*^{\text{multi}} = k)$, whereas a zero variance results in a distribution which is equal to one for the class corresponding to the highest predictive mean.

2.6.10 Hyperparameter Estimation

As we have seen in Section 2.6.1 and especially in Figures 2.5 and 2.6, the hyperparameters $\boldsymbol{\eta}$, such as the length-scale parameter γ of the Gaussian kernel, have a huge impact on the resulting regression function. For SVM approaches, we explained the cross-validation technique to tune hyperparameters. However, the greedy search in the parameter space is impractical especially for a high number of parameters. Due to the well-defined probabilistic framework, hyperparameter optimization for GP regression or classification is straightforward using maximum likelihood and maximum a posteriori estimation as explained in Section 2.2.1. The model parameters $\boldsymbol{\theta}$ that have to be found are the hyperparameters $\boldsymbol{\eta}$ of the kernel function, *i.e.* $\boldsymbol{\theta} = \boldsymbol{\eta}$. In the following, we restrict ourselves to GP regression. Hyperparameter estimation for Laplace approximation is quite similar and is described in detail in Rasmussen and Williams (2005, Section 5.5). The maximum likelihood approach to estimate $\boldsymbol{\eta}$ is as follows:

$$\hat{\boldsymbol{\eta}}^{\text{ML}} = \operatorname*{argmax}_{\boldsymbol{\eta}} \; p(\boldsymbol{y} \,|\, \mathbf{X}, \boldsymbol{\eta}) \tag{2.110}$$

$$= \operatorname*{argmin}_{\boldsymbol{\eta}} \; -\log p(\boldsymbol{y} \,|\, \mathbf{X}, \boldsymbol{\eta}) \tag{2.111}$$

$$= \operatorname*{argmin}_{\boldsymbol{\eta}} \; \frac{1}{2} \log\det\left(\tilde{\mathbf{K}}_{\boldsymbol{\eta}}\right) + \frac{1}{2}\, \boldsymbol{y}^T\, \tilde{\mathbf{K}}_{\boldsymbol{\eta}}^{-1}\, \boldsymbol{y} \;\; , \tag{2.112}$$

where we used the fact that $\boldsymbol{y} \sim \mathcal{N}\left(\mathbf{0}, \tilde{\mathbf{K}}_{\boldsymbol{\eta}}\right)$ with $\tilde{\mathbf{K}}_{\boldsymbol{\eta}} = \mathbf{K}_{\boldsymbol{\eta}} + \sigma_{\varepsilon}^2 \cdot \mathbf{I}$ and skipped additive constants. The additional subscript of the kernel matrix emphasizes its dependence on the hyperparameters.

The optimization problem (2.112) can be solved by gradient-based nonlinear optimization techniques, which require the exact gradient of the objective. Calculating the gradient components is done by applying several multi-dimensional derivation rules which can be found in Petersen and Pedersen (2008):

$$\begin{aligned}\frac{\partial}{\partial \eta_k}\left(-\log p(\boldsymbol{y} \mid \mathbf{X}, \boldsymbol{\eta})\right) &= \frac{1}{2}\mathrm{trace}\left(\tilde{\mathbf{K}}_{\boldsymbol{\eta}}^{-1}\frac{\partial}{\partial \eta_k}\tilde{\mathbf{K}}_{\boldsymbol{\eta}}\right) \\ &\quad - \frac{1}{2}\boldsymbol{y}^T\tilde{\mathbf{K}}_{\boldsymbol{\eta}}^{-1}\left(\frac{\partial}{\partial \eta_k}\tilde{\mathbf{K}}_{\boldsymbol{\eta}}\right)\tilde{\mathbf{K}}_{\boldsymbol{\eta}}^{-1}\boldsymbol{y} \qquad (2.113) \\ &= \frac{1}{2}\mathrm{trace}\left(\left(\tilde{\mathbf{K}}_{\boldsymbol{\eta}}^{-1} - \boldsymbol{\alpha}\boldsymbol{\alpha}^T\right)\frac{\partial}{\partial \eta_k}\tilde{\mathbf{K}}_{\boldsymbol{\eta}}\right) \quad , \qquad (2.114)\end{aligned}$$

We used the coefficients $\boldsymbol{\alpha} = \tilde{\mathbf{K}}_{\boldsymbol{\eta}}^{-1}\boldsymbol{y}$. The derivative of the marginal likelihood is reduced to the derivative of the kernel matrix with respect to a single hyperparameter η_k. For example, in case of the Gaussian kernel (Eq. (2.41)) and the hyperparameter $\eta = \gamma$, we have:

$$\begin{aligned}\left(\frac{\partial}{\partial \gamma}\tilde{\mathbf{K}}_{\boldsymbol{\eta}}\right)_{i,j} &= -\left|\left|\boldsymbol{x}_i - \boldsymbol{x}_j\right|\right|^2 \; K_{ij} \\ &= -\left|\left|\boldsymbol{x}_i - \boldsymbol{x}_j\right|\right|^2 \exp\left(-\gamma\left|\left|\boldsymbol{x}_i - \boldsymbol{x}_j\right|\right|^2\right) \quad , \qquad (2.115)\end{aligned}$$

which can be computed without additional computational cost by storing the pairwise distances already needed to compute the kernel matrix.

Hyperparameter Optimization for Multi-class Classification Hyperparameter optimization for the one-vs-all approach (Section 2.6.9) can be done by joint optimization of hyperparameters for all involved binary problems (Lawrence et al., 2004). Let $\boldsymbol{y}^k$ be the vector of all binary labels derived for the task of discriminating between the class $y = k$ and all remaining classes. If we assume that all binary tasks share the same hyperparameters but are otherwise independent, the objective function of the hyperparameter optimization is simply the

sum of all binary negative logarithmic likelihoods as given in Eq. (2.112):

$$\hat{\boldsymbol{\eta}}^{\mathrm{ML}} = \underset{\boldsymbol{\eta}}{\operatorname{argmax}}\ p(\boldsymbol{y}^1, \ldots, \boldsymbol{y}^M \mid \mathbf{X}, \boldsymbol{\eta}) \tag{2.116}$$

$$= \underset{\boldsymbol{\eta}}{\operatorname{argmin}}\ -\sum_{k=1}^{M} \log p(\boldsymbol{y}^k \mid \mathbf{X}, \boldsymbol{\eta}) \tag{2.117}$$

$$= \underset{\boldsymbol{\eta}}{\operatorname{argmin}}\ \frac{M}{2} \log\det\left(\tilde{\mathbf{K}}_{\boldsymbol{\eta}}\right) + \frac{1}{2}\sum_{k=1}^{M} \left(\boldsymbol{y}^k\right)^T\ \tilde{\mathbf{K}}_{\boldsymbol{\eta}}^{-1}\ \boldsymbol{y}^k \tag{2.118}$$

$$= \underset{\boldsymbol{\eta}}{\operatorname{argmin}}\ \frac{M}{2} \log\det\left(\tilde{\mathbf{K}}_{\boldsymbol{\eta}}\right) + \frac{1}{2}\operatorname{trace}\left(\tilde{\mathbf{K}}_{\boldsymbol{\eta}}^{-1} \sum_{k=1}^{M} \boldsymbol{y}^k \left(\boldsymbol{y}^k\right)^T\right) . \tag{2.119}$$

An equivalent idea can be applied to Laplace approximation and is demonstrated in Rodner et al. (2010) and further analyzed in Section 5.8.

Implementation Details As optimization techniques, we utilize the conjugate gradients methods used in the MATLAB source code accompanying the textbook of Rasmussen and Williams (2005) or the trust-region optimizer presented in Bajramovic (2004). In our experiments, we observed that both minimizers have a similar performance.

The noise variance is sometimes also considered as a hyperparameter which is optimized with the above method. However, in our implementation we choose the noise variance σ_ε^2 adaptively. We iteratively increase the value of $\sigma_\varepsilon \in \{0, 10^{-8}, 10^{-7}, 10^{-6}, \ldots\}$ until the Cholesky decomposition of the kernel matrix can be calculated ensuring its positive definiteness and well-conditioning.

Chapter 3

Learning with Few Examples

The main theoretical part of this work is presented in the following chapter. After defining transfer learning in mathematical terms, we show how to extend random decision forests to a transfer technique in two different ways (Section 3.2 and Section 3.3). The suitability of a novel Bayesian model utilizing Gaussian process (GP) priors for knowledge transfer, especially within heterogeneous environments, is demonstrated in Section 3.4. The last section concentrates on one-class classification with GP priors (Section 3.5).

3.1 Problem Formulation of Transfer Learning

In contrast to traditional machine learning, transfer learning methods exploit learning data from auxiliary tasks or categories. Most of the work on transfer learning and also a large part of this thesis concentrates on transfer learning of binary classification tasks. In addition to the training dataset $\mathcal{D}^\tau$ of the target task, we have complete access to training sets $\mathcal{D}_1^\mathcal{S}, \ldots, \mathcal{D}_J^\mathcal{S}$ of J support tasks. For example, let us consider the task of learning a new animal category and building a classifier that decides whether an animal of the new type is present in an image. Training data $\mathcal{D}^\tau$ would consist of positive and negative examples, *i.e.* images displaying the animal category and images corresponding to some background category. A learning set $\mathcal{D}_i^\mathcal{S}$ of a support task i comprises positive and negative examples of a related or visually similar animal category.

Let us give a more formal definition of the described transfer learning scenario:

Definition 3.1 (Inductive Transfer Learning) *Let $\mathcal{D}^\tau \in (\mathcal{X} \times \mathcal{Y})^{\tilde{n}}$ be given as a training set of a* target task τ. *Furthermore, a collection of training sets $\mathfrak{D}^\mathcal{S} = \left(\mathcal{D}_1^\mathcal{S}, \ldots, \mathcal{D}_J^\mathcal{S}\right)$ is given corresponding to J other tasks, referred to as* support tasks. *The goal of a* transfer learner *or* transfer learning algorithm *is to infer the unknown relationship between inputs $\boldsymbol{x} \in \mathcal{X}$ and outputs $y \in \mathcal{Y}$ of the target task using all information available in $\mathcal{D}^\tau$ and $\mathfrak{D}^\mathcal{S}$. In probabilistic terms, this goal can be stated as estimating the posterior $p(y_* \,|\, \boldsymbol{x}_*, \mathcal{D}^\tau, \mathfrak{D}^\mathcal{S})$ of the output y_* of a new example $\boldsymbol{x}_*$ corresponding to the target task τ.*

In contrast to multitask learning, we only want to learn the target task and not all tasks in combination. In general, the input spaces and the output spaces of the target task and the support tasks could differ. However, the main part of this thesis focuses on binary classification tasks with $\mathcal{Y} \in \{-1, 1\}$ and equivalent input spaces, *i.e.* the same set of features is used. We refer to this standard scenario as *binary transfer learning*. A trivial transfer learning strategy is to use all training examples of the support tasks and use them directly as additional examples of the target task. Whereas this strategy works well if the tasks are nearly identical, it fails if the relationship between the tasks is more complex and only partial.

We can formalize inductive transfer learning by gathering all information that can be transferred from support tasks to the target task into a random variable $\boldsymbol{\theta} \in \Theta$, *i.e.* the variable $\boldsymbol{\theta}$ represents the information used to train the target task in addition to the target training examples $\mathcal{D}^\tau$. We refer to it as the *transfer information*. Using such a model, we directly arrive at

$$p(y_* \,|\, \boldsymbol{x}_*, \mathcal{D}^\tau, \mathfrak{D}^\mathcal{S}) = \int_\Theta p(y_* \,|\, \boldsymbol{x}_*, \mathcal{D}^\tau, \boldsymbol{\theta}) \cdot p(\boldsymbol{\theta} \,|\, \mathcal{D}^\tau, \mathfrak{D}^\mathcal{S}) \, d\boldsymbol{\theta} \ . \tag{3.1}$$

The posterior $p(y_* \,|\, \boldsymbol{x}_*, \mathcal{D}^\tau, \boldsymbol{\theta})$ is a probabilistic model provided by the classification techniques presented in Chapter 2. The only difference is the additional use of the transfer information $\boldsymbol{\theta}$, which can be considered as an additional model parameter. The prior $p(\boldsymbol{\theta} \,|\, \mathcal{D}^\tau, \mathfrak{D}^\mathcal{S})$ of the transfer information depends on all training data provided.

Whereas the model presented here in a general manner is directly used by our GP transfer learning approach presented in Section 3.4, our random decision forest approaches described in Section 3.2 and Section 3.3, which are

referred to as regularized tree and feature relevance method, use different kinds of additional model assumptions. Both approaches assume that the transfer information is only obtained from support tasks rather than in combination with the information in the target training set. The feature relevance approach and the binary classification version of the regularized tree method also rely on one single support task selected manually in advance or chosen according to some task similarities like those proposed in Section 3.4.4.

Besides transfer learning between several binary classification tasks, the regularized tree method has the advantage of being able to tackle problems in a transfer learning domain involving multi-class classification problems. In a *multi-class transfer learning*[1] setting, we consider one multi-class classification task, such as discriminating between different animal images. Learning such a task is of course difficult if there is one single class τ with only a small number of training images. The aim of multi-class transfer learning is to exploit class similarities or relationships given in advance to the learner, such as knowing that a specific animal class is visually similar to another one. We restrict ourselves to a scenario in which only one target class and several related support classes are specified. This corresponds to online learning on a category level, *i.e.* new classes are learned sequentially by exploiting similarities to previous already known classes.

Definition 3.2 (Multi-class Transfer Learning) *Let $\mathcal{D} \in (\mathcal{X} \times \{1, \ldots, M\})^n$ be a training set of a multi-class classification task with M classes. Furthermore, let $\tau \in \{1, \ldots, M\}$ be a single class which is related to a set $\mathcal{S} \subseteq \{1, \ldots, M\}$ of J related classes. We refer to τ as the* target class *and to $\mathcal{S}$ as the set of* support classes*. The goal of a* multi-class transfer learner *or* multi-class transfer learning algorithm *is to solve a multi-class classification task by exploiting the relationships given between classes, e.g. visual similarity. We define $\mathcal{D}^k \subset \mathcal{D}$ to be the set of all training examples of class k and use $\mathcal{D}^{\mathcal{S}}$ as a notation of the training data belonging to all support classes in $\mathcal{S}$.*

In contrast to the definition of inductive transfer learning, we are using the term "class" instead of "task" to emphasize that we are working in one single multi-class classification task. The difficulty of this setting is that the learner despite transferring knowledge from related classes also has to discriminate between the target and support classes.

[1] Unfortunately, there seems to be no established term to express this special scenario of transfer learning.

The trivial method of using all available training examples of the support classes as training examples of the target class is not available in this scenario. Due to this reason, there is a trade-off between transferring all information leading to non-separable classes and not transferring any information leading to a poor performance due to the lack of training data.

A hierarchical classification algorithm using a hierarchy given as prior information to the learner can be seen as solving a related problem, because it also exploits class similarities indirectly given by the grouping (Zweig and Weinshall, 2007). Another example of multi-class transfer learning is the congealing approach of Miller et al. (2000), which has been already discussed in Section 1.3.1.

3.2 Transfer with Regularized Decision Trees

In the following, we present our approach to multi-class transfer learning, which was published in Rodner and Denzler (2008) and Rodner and Denzler (2011). We are dealing with a multi-class classification task that contains a target class τ with few training examples and the goal is to improve the recognition performance by using prior similarities between the categories.

Our approach is based on random decision forests (RDF) as presented in Section 2.3. With few training examples a decision tree classifier gives poor results, because the posterior distribution of the target class is modeled by a discrete histogram that determines the posterior probability (Section 2.3.1.1). If we only have a small amount of training data, the histograms are likely to be very sparse, *i.e.* zero probability is assigned to nearly all parts of the input space. We propose to re-estimate all probabilities using prior information obtained from all support classes.

The method focuses on multi-class transfer learning, although it can be easily applied in the standard inductive transfer learning domain as will be presented in Section 3.2.6. The algorithm can be applied to each tree of the forest individually, therefore, the details of our method are explained using a single decision tree. Additionally, it does not depend on the method used to build the trees, however, we stick to the randomized learning procedure given in Section 2.3.2. We transfer two different types of information: a discriminative tree structure and a prior distribution of leaf probabilities. Figure 3.1 summarizes the main steps of our approach.

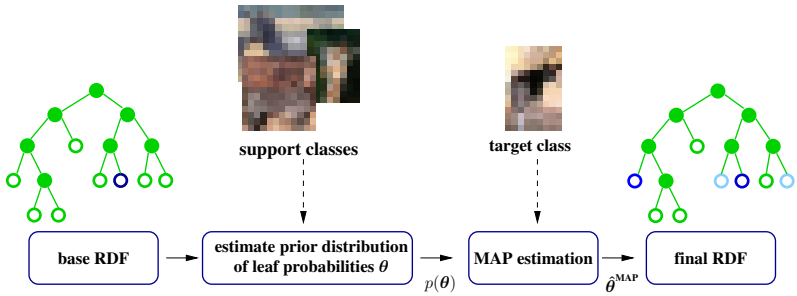


Figure 3.1: Outline of the regularized tree approach presented in Section 3.2: We build a random decision forest (RDF) using all training data, in which leaf probabilities of a target class are re-estimated using a prior distribution obtained from a set of support tasks. Shaded nodes in the decision tree are leaves with non-zero posterior probability of the target class.

3.2.1 Transferring the Decision Tree Structure

The selection of discriminative features using few examples is a highly ill-posed problem, especially in high-dimensional spaces. Therefore, we construct a discriminative tree structure using the available training data of all M classes. Another option is to use all training examples except of the target class τ, which allows using the transfer approach in a class-level online learning scenario. A fixed tree structure determines a decomposition of the input space into several cells. Thus, we transfer the information which parts of the input space are likely to have uniform properties concerning the object categories.

The concept of reusing decision trees is also utilized in Hoiem et al. (2007) and Lepetit et al. (2005) to recycle features and to reduce computation time. Additionally, the assumption of shared discriminative features (or weak learners) is similar to the use of shared features in the work of Torralba et al. (2007).

3.2.2 Transfer of Leaf Probabilities

Although decision tree approaches belong to discriminative classification techniques, they are closely related to individual density estimation. The tree structure is a partitioning of the whole input space into several cells represented by m_ℓ leaf nodes ϑ_i. This concept corresponds to an approximation of the posterior

distributions of the individual classes using a piecewise constant density or discrete probability distribution. Let $\vartheta(\boldsymbol{x})$ be the leaf node reached by the path of an example $\boldsymbol{x}$. The probability of an example of class k reaching leaf node ϑ_i is given by $p(\vartheta(\boldsymbol{x}) = \vartheta_i \,|\, y = k)$, which we refer to as leaf probabilities in the following. Additionally, we use $\boldsymbol{t}^{(k)} \in [0,1]^{m_\ell}$ to collect all leaf probabilities conditioned on class k.

Traditional decision tree algorithms use maximum likelihood (ML) estimates of a multinomial distribution to estimate the probability of each cell induced by ϑ_i:

$$t_i^{(k)} = \frac{|\left\{(\boldsymbol{x}, k) \in \mathcal{D}^k \,|\, \vartheta(\boldsymbol{x}) = \vartheta_i\right\}|}{|\mathcal{D}^k|} \quad . \tag{3.2}$$

Note that the numerator represents the number of examples of class k reaching a node ϑ_i during the training step. It should be noted that with a careful implementation of decision trees, which store those unnormalized values instead of the posterior probability, a complex recursive computation of leaf probabilities as presented in Rodner and Denzler (2008) is not necessary.

It is obvious that with only a few training examples $\boldsymbol{x} \in \mathcal{D}^\tau$ of the target class τ, the vector $\boldsymbol{t}^{(\tau)}$ is sparse and is unable to provide a good approximation of the underlying distribution. In the following, we use $\boldsymbol{\theta} \stackrel{\text{def}}{=} \boldsymbol{t}^{(\tau)}$ to refer to this crucial part of the model. The overall goal of our approach is to re-estimate $\boldsymbol{\theta}$ by using *maximum a posteriori estimation (MAP estimation)*, which leads to a smoother solution $\hat{\boldsymbol{\theta}}^{\text{MAP}}$:

$$\hat{\boldsymbol{\theta}}^{\text{MAP}} = \operatorname*{argmax}_{\boldsymbol{\theta} \in \Theta} p(\mathcal{D}^\tau \,|\, \boldsymbol{\theta})\; p(\boldsymbol{\theta} \,|\, \mathfrak{D}^{\mathcal{S}}) \quad , \tag{3.3}$$

where we use a prior on $\boldsymbol{\theta}$ obtained from the training data $\mathfrak{D}^{\mathcal{S}}$ of all support classes. Since the leaves of a decision tree induce a partitioning of the input space in disjoint subsets

$$\mathcal{X}^{(i)} = \{\boldsymbol{x} \in \mathcal{X} | \vartheta(\boldsymbol{x}) = \vartheta_i\} \quad , \tag{3.4}$$

each instance of the parameter vector $\boldsymbol{\theta}$ is a discrete multinomial distribution and fulfills $\sum_i \theta_i = 1$. For this reason, any suitable distribution of discrete distributions can be used to model the prior distribution $p(\boldsymbol{\theta} \,|\, \mathfrak{D}^{\mathcal{S}})$.

3.2.3 Constrained Gaussian Prior

We propose to use a *constrained Gaussian distribution (CGD)* which is a simple family of parametric distributions and can serve as an alternative to a standard Dirichlet distribution. For all vectors[2] $\boldsymbol{\theta} \in \mathbb{R}^{m_\ell}_{\geq 0}$ with non-negative entries, the density is defined as

$$p(\boldsymbol{\theta} \,|\, \mathfrak{D}^{\mathcal{S}}) \propto \mathcal{N}(\boldsymbol{\theta} \,|\, \boldsymbol{\mu}, \sigma^2 \cdot \mathbf{I}) \;\cdot\; \delta\left(1 - \sum_{i=1}^{m_\ell} \theta_i\right) \;. \tag{3.5}$$

The factor involving the discrete delta function $\delta(\cdot)$, with $\delta(0) = 1$ and $\forall x \neq 0 : \delta(x) = 0$, is essential to ensure that the support of the density function is the simplex of all feasible discrete distributions. The use of $\sigma^2\mathbf{I}$ as a covariance matrix is an additional assumption that will be useful in deriving an efficient MAP estimation algorithm (Section 3.2.4).

This simple model allows us to estimate hyperparameters $\boldsymbol{\mu}$ and σ^2 in the usual way. Due to the reason that the simplex is a convex set, the mean vector $\boldsymbol{\mu}$ can be estimated analogously to a non-constrained Gaussian. In our application of this principle to decision trees, $\boldsymbol{\mu}$ is estimated using the leaf probabilities of all J support classes:

$$\boldsymbol{\mu} = \frac{1}{J} \sum_{j \in \mathcal{S}} \boldsymbol{t}^{(j)} \;. \tag{3.6}$$

Our choice to model the unknown distribution by a Gaussian parametric family is mostly due to practical computational considerations rather than theoretical results. One could argue that using a symmetric Dirichlet prior, which is the proper conjugate prior of a multinomial, leads to the same set of parameters as a CGD. In our application of regularized trees, we expect a symmetric Dirichlet prior to yield similar results. Our motivation to use a CGD is the scientifically interesting fact that even without a conjugate prior, one can derive a simple inference method using an easy to solve one-dimensional optimization problem. An investigation and analysis of other parametric distributions and more sophisticated priors could be an interesting topic for future research.

[2] Although we define the prior in a general manner, the connection to our application to decision trees is highlighted by using a m_ℓ-dimensional vector with m_ℓ denoting the number of leaf nodes in a decision tree.

3.2.4 MAP Estimation with a Constrained Gaussian Prior

MAP estimation using complex parametric distributions often requires nonlinear optimization techniques. In contrast to these approaches, we briefly show that by using our constrained Gaussian as a prior of a multinomial distribution, it is possible to derive a closed-form solution of the global optimum depending on a single Lagrange multiplier.

We start by writing the objective function of the MAP estimation as a Lagrange function of our simplex constraint and the posterior:

$$L(\boldsymbol{\theta}, \lambda) = \log\left(p(\mathcal{D}^\tau \mid \boldsymbol{\theta})\; p(\boldsymbol{\theta} \mid \mathfrak{D}^{\mathcal{S}})\right) + \lambda \left(\sum_{i=1}^{m_\ell} \theta_i - 1\right) \ .$$

The likelihood has a simple multinomial form and depends on a discrete histogram $\boldsymbol{c} = (c_i)_{i=1}^{m_\ell}$ representing the number of samples of each component:

$$p(\mathcal{D}^\tau \mid \boldsymbol{\theta}) \propto \prod_{i=1}^{m_\ell} (\theta_i)^{c_i} \quad . \tag{3.7}$$

In our application to leaf probabilities of decision trees, the absolute number of examples of class τ reaching a node is used. With the CGD prior in Eq. (3.5) we obtain the overall objective function

$$\sum_{i=1}^{m_\ell} \left(c_i \log(\theta_i) - \frac{1}{2\sigma^2}(\theta_i - \mu_i)^2 + \lambda\theta_i\right) - \lambda \ .$$

This objective function is convex and therefore has a unique solution. Setting the gradient $\left(\frac{\partial L}{\partial \theta_i}\right)(\boldsymbol{\theta}, \lambda)$ to zero leads to m_ℓ independent equations

$$0 = \frac{c_i}{\theta_i} - \frac{1}{2\sigma^2} \cdot 2 \cdot (\theta_i - \mu_i) + \lambda \ . \tag{3.8}$$

Note that we get a non-informative prior for $\sigma^2 \to \infty$, which reduces MAP to ML estimation. With positive discrete probabilities ($\theta_i > 0$), it is possible to obtain a simple quadratic equation in θ_i:

$$0 = (\theta_i)^2 + \theta_i\,(-\mu_i - \lambda\sigma^2) - \sigma^2 c_i \ . \tag{3.9}$$

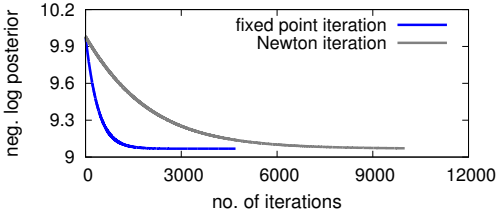


Figure 3.2: Comparison of the convergence of the Newton method and a simple fixed point iteration for MAP estimation using a constrained Gaussian prior as presented in Section 3.2.4.

A stationary point with $\theta_i = 0$ is only possible with $c_i = 0$ or $\sigma^2 \to 0$, which is also reflected by the above equation. Therefore, the optimization problem has only a single non-negative solution depending on λ:

$$\theta_i = \frac{\mu_i + \lambda\sigma^2}{2} + \sqrt{\left(\frac{\mu_i + \lambda\sigma^2}{2}\right)^2 + \sigma^2 c_i} \ . \tag{3.10}$$

This solution depends on the Lagrange multiplier, for which an optimal value can be found using a simple fixed point iteration derived from the constraints:

$$\lambda^{(j+1)} = \frac{1}{m_\ell \cdot \sigma^2}\left(1 - 2\sum_{i=1}^{m_\ell}\sqrt{\left(\frac{\mu_i + \lambda^{(j)}\sigma^2}{2}\right)^2 + \sigma^2 c_i}\right) \ . \tag{3.11}$$

As an initial value, it is possible to use the optimal Lagrange multiplier derived for the case of no prior knowledge and maximum likelihood estimation. Figure 3.2 shows the convergence of our technique compared to that of a Newton iteration, which converges much slower than our simple recursion formula given in Eq. (3.11).

3.2.5 Building the Final Tree for Multi-class Transfer Learning

In contrast to previous work, which often concentrate on the binary case (*e.g.* Fei-Fei et al., 2006), regularized trees are suitable for multi-class classification

problems. Given the leaf probabilities $\boldsymbol{t}^{(k)}$ and prior probabilities $p(y = k)$ for each class $k \in \{1, \ldots, M\}$, we can easily calculate the needed posterior probabilities for each class by utilizing Bayes' law:

$$p(y = k \,|\, \vartheta(\boldsymbol{x}) = i) = \frac{p(\vartheta(\boldsymbol{x}) = i \,|\, y = k) \cdot p(y = k)}{\sum_{k'=1}^{M} \left(p(\vartheta(\boldsymbol{x}) = i \,|\, y = k') \cdot p(y = k')\right)} \tag{3.12}$$

$$= \frac{t_i^{(k)} \cdot p(y = k)}{\sum_{k'=1}^{M} \left(t_i^{(k')} \cdot p(y = k')\right)} \,. \tag{3.13}$$

All machine learning approaches using the multi-class transfer learning paradigm have to cope with a common issue: transferring knowledge from support classes can lead to confusion with the target class. For example, using prior information from *camel* images to support the class *dromedary* enables us to transfer shared features like fur color or head appearance. However, we have to use additional features (*e.g.* shape information) to discriminate between both categories.

To solve this problem, we propose to build additional discriminative levels of the decision tree after MAP estimation of leaf distributions. Starting from a leaf node i with non-zero posterior probability $p(y = k \,|\, \vartheta(\boldsymbol{x}) = \vartheta_i)$, the tree is further extended by the randomized training procedure described in Section 2.3.2. The training data consists of all examples of the target class and examples of all support classes that reached leaf ϑ_i. All of the training examples are additionally weighted by the posterior probabilities of the corresponding classes. This technique allows us to find new discriminative features especially between the target class and the support classes.

3.2.6 Building the Final Tree for Binary Transfer Learning

Transfer learning for binary classification relies on a set of support tasks that try to separate a class k and a background class $\mathcal{B}$. Regularized trees can be applied straight-forwardly to this setting, if a single support classification tasks is given and the same background class is used in all tasks. Thus, we have two binary classification tasks, a target task that discriminates between τ and $\mathcal{B}$ and a support task discriminating between s and $\mathcal{B}$. After building a random decision forest using training data of the support task, *i.e.* all examples of s and $\mathcal{B}$, we can apply the re-estimation method as explained in Section 3.2.4 using the mean

vector $\boldsymbol{\mu} = \boldsymbol{t}^{(s)}$. Finally, the class probabilities of τ are substituted for s, and the decision tree now tries to separate between s and $\mathcal{B}$.

3.2.7 Related Work

Our work on regularized decision trees is related to the approach of Lee and Giraud-Carrier (2007). The key idea of their method is the re-usability of a decision tree structure from a related binary classification task. In contrast, we introduce a technique that also reuses estimated class probabilities in leaf nodes and performs a re-estimation based on maximum a posteriori. In general, the concept of shared priors is used in other transfer learning approaches, such as the extension of generalized linear models by Lee et al. (2007).

Another perspective of our multi-class transfer learning method offers the following formulation of decision trees: Let $\boldsymbol{w}^{(k)} \in [0,1]^{m_\ell}$ be the vector of leaf posterior probabilities of the class k as defined in Eq. (3.12). Furthermore, we make use of the function $\phi : \mathcal{X} \to \{0,1\}^{m_\ell}$ which returns a binary vector and indicates the leaf reached by the path of an example $\boldsymbol{x} \in \mathcal{X}$, *i.e.* $\phi_i(\boldsymbol{x}) = 1$ if and only if $\vartheta(\boldsymbol{x}) = \vartheta_i$. The resulting posterior of the label y_* of a new example $\boldsymbol{x}_*$ can now be written as:

$$p(y_* = k \,|\, \boldsymbol{x}_*, \mathcal{D}) = \sum_{i=1}^{m_\ell} w_i^{(k)} \cdot \phi_i(\boldsymbol{x}) = \left(\boldsymbol{w}^{(k)}\right)^T \phi(\boldsymbol{x}) \ . \tag{3.14}$$

Hence, a decision tree classifier transforms each input and subsequently applies a linear classifier in a similar fashion as indirectly done by a kernel machine like SVM (Section 2.5) or GP-related methods (Section 2.6). An important difference is the use of a learned feature transformation ϕ inferred from the training data instead of a predefined one.

In view of our method, we indirectly place a prior on hyperplane parameters $\boldsymbol{w}^{(\tau)}$ that depends on training data of support classes. This technique, also known as *joint regularization*, is a common transfer learning strategy for multitask scenarios and for example used by Amit et al. (2007) or Argyriou et al. (2006). More information about these approaches can be found in Section 1.3.1. Using a tree-based feature transformation is also proposed by Nowak and Jurie (2007) for object identification and comparing. Transferring the tree structure and using the decision tree learned by all classes is translated to estimating and transferring the transformation ϕ, which is also the main idea of Quattoni et al. (2007) and Tang et al. (2010). Adding additional discriminative levels to a decision tree is

also proposed by Belle et al. (2008) to sequentially learn the face appearance of new people in a face identification task.

3.3 Transfer of Feature Relevance

The following section is partly based on our publication in Rodner and Denzler (2009a) and presents an approach to transfer learning of binary classification tasks by transferring feature relevance.

One problem of learning with few examples is the inability of traditional machine learning algorithms to determine relevant features from a large pool of generic features. Therefore, transferring the relevance of features from support tasks can be very helpful to increase the generalization performance. Feature relevance can be roughly defined as the usefulness of a feature value to predict the class of an object instance and it is mostly defined in terms of mutual information (Guyon et al., 2006). To give an illustrative example of our transfer idea, consider again the recognition of a new animal class. With the aid of prior knowledge from related animal classes, such as the knowledge about the importance of typical body parts like hooves, legs, and head, learning is much easier.

We concentrate on knowledge transfer between two binary classification tasks. A support task with a relatively large number of training examples and a target task with few training examples is given. We assume that support task and target task share a common set of relevant features. For this reason, probabilities of feature relevance for a support task are estimated in a preliminary step. This estimation is able to use a large number of training examples and thus yields more accurate results than an estimation using just few training examples of the target task. The estimated distribution of feature relevance can then be utilized in the construction process of a random decision forest (Section 2.3). In contrast to the regularized tree method presented in the previous section, which uses a uniform feature distribution, the prior information increases the probability of a relevant feature to be selected for the target task. Figure 3.3 provides an overview of this idea.

The remainder of this section is structured as follows: After defining feature relevance in Section 3.3.1, we present our transfer learning idea (Section 3.3.2) and explain its incorporation into randomized classifier ensembles (Section 3.3.3). How to estimate the probability of feature relevance with random decision forests is described in Section 3.3.5.

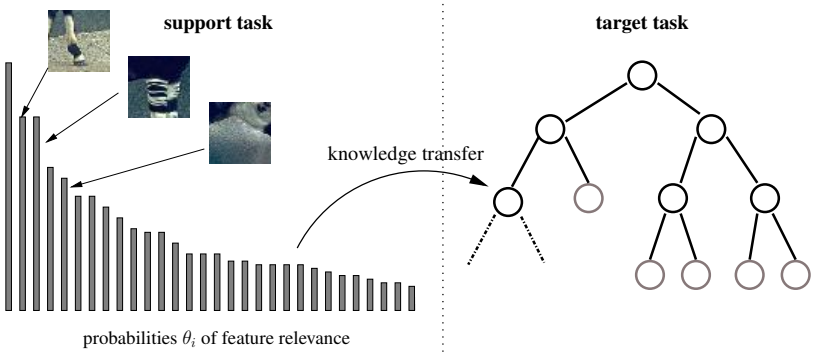


Figure 3.3: Illustration of the general principle of our approach: Feature relevance is estimated from a support task and used to regularize the feature selection in the training process (random decision forest) of a target task with few training examples. The probabilities and BoV feature visualizations are directly obtained from our image categorization task (Section 5.2).

3.3.1 Feature Relevance

Whereas the previous part of this thesis used the term feature to refer to a single component in multi-dimensional input vectors $\boldsymbol{x} \in \mathcal{X} \subseteq \mathbb{R}^D$, in the following, we use a more general definition of a feature as a function $g : \mathcal{X} \rightarrow \mathbb{R}$. Additionally, we denote the (ordered) set of all available features as $\mathcal{F}$ and $\mathcal{F}(\boldsymbol{x})$ is short for the vector containing all feature values of $\boldsymbol{x}$, *i.e.* all functions in $\mathcal{F}$ are evaluated and the results are stored in a vector. The set of features is application-specific and has to be defined in advance. If we define $\mathcal{F}$ to be the set of all projections of a D-dimensional vector, we easily see that our current definition is a generalization of the our previous usage of the term feature.

Although we might have an intuitive understanding of the term relevant feature, its mathematical definition is tricky. A common definition of a relevant feature $g \in \mathcal{F}$ is given by the following property (John et al., 1994, Definition 2): $\exists(\boldsymbol{x}', k) \in \mathcal{X} \times \{-1, 1\}$

$$p(g(\boldsymbol{x}) = g(\boldsymbol{x}'), y = k) \;\neq\; p(g(\boldsymbol{x}) = g(\boldsymbol{x}')) \cdot p(y = k) \;\;, \tag{3.15}$$

where we explicitly differentiate between the random variables $\boldsymbol{x}, y$ and their

instances $\boldsymbol{x}'$ and k. However, as pointed out by John et al. (1994), this definition is unable to capture dependencies between features. Consider a simple example with $\mathcal{X} = \{0,1\}^2$, $\mathcal{F}$ being the set of projections and $y = (x_1 \text{ xor } x_2)$. Given only one of both features, we can not say anything about the label at all. Therefore, the definition suggests that both features are not relevant. Due to this reason we have to consider sets of relevant features as done by Guyon et al. (2006).

Definition 3.3 (Surely sufficient feature set) *A set of features $\mathcal{R} \subseteq \mathcal{F}$ is said to be* surely sufficient *if and only if the following holds in general*

$$p(y \,|\, \mathcal{R}(\boldsymbol{x})) = p(y \,|\, \mathcal{F}(\boldsymbol{x})) \ . \tag{3.16}$$

Thus, informally speaking $\mathcal{R}$ extracts all information about a label y available in $\mathcal{F}$.

A surely sufficient feature subset can contain redundant features, *i.e.* features which can be removed without losing any information about the label. For example, the set of all available features is always surely sufficient. Reducing redundant features from the definition is done by defining minimal feature subsets:

Definition 3.4 (Minimal sufficient or relevant feature set) *A set of features $\mathcal{R} \subseteq \mathcal{F}$ is called* minimal sufficient *or* relevant *if and only if $\mathcal{R}$ is surely sufficient and there is no other surely sufficient feature set of smaller size.*

Knowing a relevant feature set can be very beneficial for a learner. As already elaborated in the introduction (Section 1.1), with a smaller set of features, a classifier can be learned with a lower complexity leading to an expected lower generalization error. In the next sections, we describe our method, which is able to transfer information about a relevant feature set from a support task.

3.3.2 Feature Relevance for Knowledge Transfer

Let $\mathcal{D}^{\mathcal{S}} = \{(\boldsymbol{x}_i, y_i)\}_{i=1}^n$ be a set of training examples of a given support binary classification task with $y_i \in \{-1, 1\}$ and inputs $\boldsymbol{x}_i \in \mathcal{X}$. Given few training examples a learner tends to overfit and a classification decision is often based on irrelevant or approximately irrelevant features (Guyon et al., 2006). The goal of our approach is to reduce overfitting by incorporating a prior distribution $\boldsymbol{\theta}$ on relevant features.

Our approach to transfer learning relies on the assumption that support task and target task share a set $\mathcal{R} \subseteq \mathcal{F}$ of relevant features. We therefore transfer the probability θ_i of a feature g_i to be relevant by using the training examples $\mathcal{D}^{\mathcal{S}}$ of the support task. We additionally assume that the relevance of features is independently distributed:

$$p(\tilde{\mathcal{R}} \,|\, \mathcal{F}, \mathcal{D}^{\mathcal{S}}) = \prod_{g_i \in \tilde{\mathcal{R}}} \underbrace{p(g_i \in \mathcal{R} \,|\, \mathcal{F}, \mathcal{D}^{\mathcal{S}})}_{\theta_i} \quad . \tag{3.17}$$

While we delay the estimation of $\boldsymbol{\theta}$ to Section 3.3.5, the following section shows that $\boldsymbol{\theta}$ can be used as a prior distribution on the set of possible hypotheses or models for the target task. This also highlights that our prior knowledge can be easily integrated in the concept of randomized classifier ensembles and especially in random decision forests (Section 2.3).

3.3.3 Incorporation into Randomized Classifier Ensembles

We now describe the random decision forest approach in a theoretical framework related to Bayesian model averaging (Section 2.2.2). This allows us to motivate the transfer of feature relevance as a Bayesian approach of defining a prior distribution on models or hypotheses.

The final goal is to estimate the probability of the event $y_* = 1$ that a previously unseen object instance $\boldsymbol{x}$ belongs to class 1 conditioned on the set of few training examples $\mathcal{D}^{\mathcal{T}}$ of the target task and the set of all possible features $\mathcal{F}$. As a classification model, we use an ensemble of base models h (in our case single decision trees) in the following manner:

$$p(y_* = 1 \,|\, \boldsymbol{x}_*, \mathcal{D}^{\mathcal{T}}, \mathcal{F}) = \int_{h \in H} p(y_* = 1 \,|\, \boldsymbol{x}_*, h)\; p(h \,|\, \mathcal{D}^{\mathcal{T}}, \mathcal{F})\; dh \quad . \tag{3.18}$$

The model h is often assumed to be deterministic for a given training and feature set, but there are multiple ways to sample from those sets and thus generate multiple models. One idea is the concept of bagging (Section 2.2.3) which uses random subsets of the training data. As proposed by Breiman (2001) and Geurts et al. (2006) another possibility is to use random subsets of all features (Section 2.3.2). This approach can be regarded as Bayesian model averaging and reflects our uncertainty about the set $\mathcal{R}$ of relevant features:

$$p(h \,|\, \mathcal{D}^{\mathcal{T}}, \mathcal{F}) = \sum_{\tilde{\mathcal{R}} \subseteq \mathcal{F}} p(h \,|\, \mathcal{D}^{\mathcal{T}}, \tilde{\mathcal{R}})\; p(\tilde{\mathcal{R}} \,|\, \mathcal{F}) \quad . \tag{3.19}$$

The distribution $p(\tilde{\mathcal{R}} \,|\, \mathcal{F})$ describes the probability that $\tilde{\mathcal{R}}$ is the set of relevant features. Note that we use a simplified notation which hides the fact that we are handling with binary random variables indicating the membership to $\mathcal{R}$. A base model h is deterministic given a training set and the set of relevant features:

$$p(h \,|\, \mathcal{D}^\tau, \tilde{\mathcal{R}}) = \delta\left[h - h(\mathcal{D}^\tau, \tilde{\mathcal{R}})\right] . \tag{3.20}$$

Combining all equations yields the final classification model:

$$p(y_* = 1 \,|\, \boldsymbol{x}_*, \mathcal{D}^\tau, \mathcal{F}) = \sum_{\tilde{\mathcal{R}} \subseteq \mathcal{F}} p(y_* = 1 \,|\, \boldsymbol{x}_*, h(\mathcal{D}^\tau, \tilde{\mathcal{R}}))\; p(\tilde{\mathcal{R}} \,|\, \mathcal{F}) \;\; . \tag{3.21}$$

This sum can not be computed efficiently for large input spaces, therefore, we can approximate it by simple Monte Carlo estimation:

$$p(y_* = 1 \,|\, \boldsymbol{x}_*, \mathcal{D}^\tau, \mathcal{F}) = \frac{1}{T} \sum_{t=1}^{T} p(y_* = 1 \,|\, \boldsymbol{x}_*, h(\mathcal{D}^\tau, \mathcal{R}^t)) \;\; . \tag{3.22}$$

Feature subsets $\mathcal{R}^t$ are sampled from $p(\tilde{\mathcal{R}} \,|\, \mathcal{F})$. This distribution is often assumed to be uniform and samples of only a fixed number of features $|\tilde{\mathcal{R}}|$ are used (Geurts et al., 2006), which assumes that a prior estimate of $|\mathcal{R}|$ is given. In contrast, we can apply the idea of our transfer learning technique that was described at the beginning of Section 3.3.2. Instead of using a uniform distribution $p(\tilde{\mathcal{R}} \,|\, \mathcal{F})$ we can use the probabilities $\theta_i = p(g_i \in \mathcal{R} \,|\, \mathcal{F}, \mathcal{D}^\mathcal{S})$ obtained from the support task. This prior information reduces the uncertainty of the learner about an optimal set of relevant features.

3.3.4 Application to Random Decision Forests

The presented transfer learning framework can directly be applied to random decision forests. As already presented in Section 2.3.2, building a tree is done by iteratively splitting the training set with the most informative base classifier. The selection of a classifier is done by choosing the base classifier with the highest gain in information from a random fraction of features $\mathcal{R}_v$ and possible thresholds.

Applying our transfer learning idea is straightforward by sampling the random subset of features $\mathcal{R}_v$ from the prior distribution $p(\mathcal{R} \,|\, \mathcal{F}, \mathcal{D}^\mathcal{S})$. Note that

in contrast to our theoretical framework, the selection of a random subset of features is performed in each node rather than a single time for the whole decision tree. This fact is also highlighted by the illustration in Figure 3.3. Relevant features are sampled from the distribution $\boldsymbol{\theta}$ in each split node during the training process.

3.3.5 Estimating Feature Relevance

As pointed out by Rogers and Gunn (2005), the use of ensembles of decision trees allows providing robust estimates of feature relevance that also incorporate dependencies between features. Our technique is similar to their method which uses a modified average mutual information between a feature and the class variable y in each inner node.

The first step to estimate underlying feature relevance of the support task is the training of a random decision forest with all training examples. Afterward, we count the number of times c_i a feature g_i is used in a split node. A feature with a high occurrence c_i is likely to be relevant for this task. To obtain the final vector $\boldsymbol{\theta}$ of feature relevance, we use maximum a posteriori estimation:

$$\hat{\boldsymbol{\theta}}^{\text{MAP}} = \underset{\boldsymbol{\theta}}{\operatorname{argmax}}\, p(\mathcal{D}^{\mathcal{S}} \mid \boldsymbol{\theta})\; p(\boldsymbol{\theta} \mid \boldsymbol{\beta}) \tag{3.23}$$

$$= \underset{\boldsymbol{\theta}}{\operatorname{argmax}} \left(\prod_i \theta_i^{c_i} \right)\; p(\boldsymbol{\theta} \mid \boldsymbol{\beta}) \;, \tag{3.24}$$

with $\boldsymbol{\beta}$ being the hyperparameter of a Dirichlet prior of $\boldsymbol{\theta}$

$$p(\boldsymbol{\theta} \mid \boldsymbol{\beta}) \propto \prod_{i=1} \theta_i^{\beta_i - 1} \;. \tag{3.25}$$

We use uniform hyperparameters, *i.e.* $\forall i : \beta_i = \beta$ in the following. Without this prior distribution, the optimal $\boldsymbol{\theta}$ is the normalized vector $\boldsymbol{c}$ of all counts. The prior distribution can be thought of as a smoothing term that prevents zero probability of relevance for some features. This is theoretically important if there is a feature g_i completely irrelevant for the support task but discriminative for the target task. The work of Kemmler and Denzler (2010) evaluates our method and compares it to other relevance techniques using random decision forests. They show that by incorporating the mutual information and the depth of a node as additional weighting factors, one can get a better performance in a feature

selection task. However, the advantage of our method is the direct availability of a well-defined probability distribution in addition to a relevance ranking, which is an essential property to apply our transfer learning idea.

In Section 5.2.2, we evaluate the influence of the parameter β. Surprisingly it turns out that in our setting a flat prior distribution ($\beta = 1$) is sufficient.

3.3.6 Related Work

Shared relevant features and class boundaries are exploited in the work of Torralba et al. (2007). They develop a boosting technique that jointly learns several binary classification tasks similar to the combined boosting idea of Levi et al. (2004). Lee et al. (2007) transfer feature relevance as a prior distribution on a weight vector in a generalized linear model. Our work is similar to their underlying idea of transferring feature relevance. In contrast, prior knowledge in our work is defined using the probability of a feature to be relevant instead of a prior distribution on a specific model parameter. Our approach additionally offers to use a state-of-the-art classifier directly.

In general, our method can be thought of as a generalization of a very common transfer learning technique that selects a subset of features, which achieved a good performance in support tasks, and uses the same fixed selection for the target task. This strategy is often indirectly applied in the scientific community in general. If a large set of papers shows the suitability of a certain set of features, *e.g.* bag of visual words as presented in Section 4.2, the same set of features is likely to be used in following papers trying to solve other but related applications. The advantage of our approach is that features are not selected in a deterministic manner and the learner of a target task is not restricted to the subset of features relevant in support tasks. Therefore, features can be selected which are very specific for the given target task and which are not shared with the support tasks.

3.4 Non-parametric Transfer with Gaussian Processes

The following approach has been published in Rodner and Denzler (2010) and offers the possibility to perform transfer learning of binary classification task within heterogeneous learning scenarios. The method is based on classification

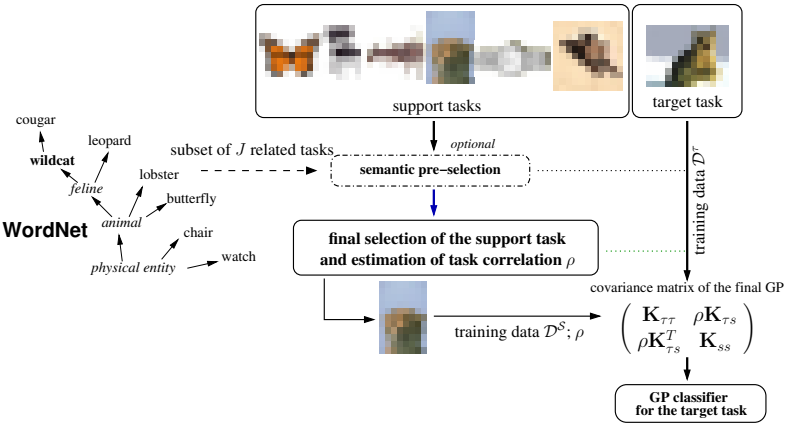


Figure 3.4: Basic outline of the proposed transfer learning approach: Semantic similarities between categories and leave-one-out estimates are utilized to select a support task, which is used to transfer knowledge to a target task with dependent Gaussian processes.

and regression with Gaussian processes (GP) as presented in Section 2.6. We use dependent Gaussian process priors, as studied by Chai (2009) and Bonilla et al. (2008) for regression and show how to utilize them for classification tasks by extending their framework in several ways. Dependent GP priors allow us to efficiently transfer the information contained in the training data of a support classification task in a non-parametric manner by using a combined (kernel) covariance function. The amount of information transferred is controlled by a single parameter estimated automatically, which allows moving gradually from independent to complete combined learning.

Additionally, we handle the case of heterogeneous tasks, where the set of available support tasks also includes unrelated categories that do not contain any valuable information for the target task. Similar to Tommasi and Caputo (2009), we utilize efficient leave-one-out estimates to select a single support classification task. We also show how to use category-level similarities estimated with WordNet (Pedersen et al., 2004) to improve this selection. The basic steps of our approach are illustrated in Figure 3.4. The framework itself is not restricted to image categorization and can be applied to any binary transfer learning scenario.

3.4.1 Dependent Gaussian Processes

As we have seen in Section 2.6 the GP framework makes use of a kernel function that determines the correlation between two function values. Dependent Gaussian processes as proposed by Chai (2009) allow us to model the correlations between P tasks and how the given tasks are related within a transfer learning scenario. For each task j, we have a latent function f^j which is assumed to be sampled from a GP prior $\mathcal{GP}(0, K^{\mathcal{X}})$. The key idea is that these functions are not assumed to be independent samples, which allows us to transfer knowledge between latent functions. Thus, a combined latent function $f((j, \boldsymbol{x})) = f_j(\boldsymbol{x})$ is used which is a single sample of a GP prior with a suitable kernel function modeled by:

$$K((j, \boldsymbol{x}), (j', \boldsymbol{x}')) = K^f_{jj'} \cdot K^{\mathcal{X}}(\boldsymbol{x}, \boldsymbol{x}') \ , \tag{3.26}$$

with $\mathbf{K}^f \in \mathbb{R}^{P \times P}$ used to model the correlations of the task-specific latent functions and $K^{\mathcal{X}}$ being a base kernel function measuring the similarities of input examples. The underlying assumption of the factorized model in Eq. (3.26) is that the latent functions are correlated in the same manner in the whole input space $\mathcal{X}$.

Knowledge transfer with dependent GPs can also be motivated theoretically with a decomposition of the latent function into an average latent function $\bar{f} \sim \mathcal{GP}(0, K^{\mathcal{X}})$ shared by all tasks and independent latent functions $\tilde{f}^j \sim \mathcal{GP}(0, K^{\mathcal{X}})$ (Pillonetto et al., 2010, assumption 4):

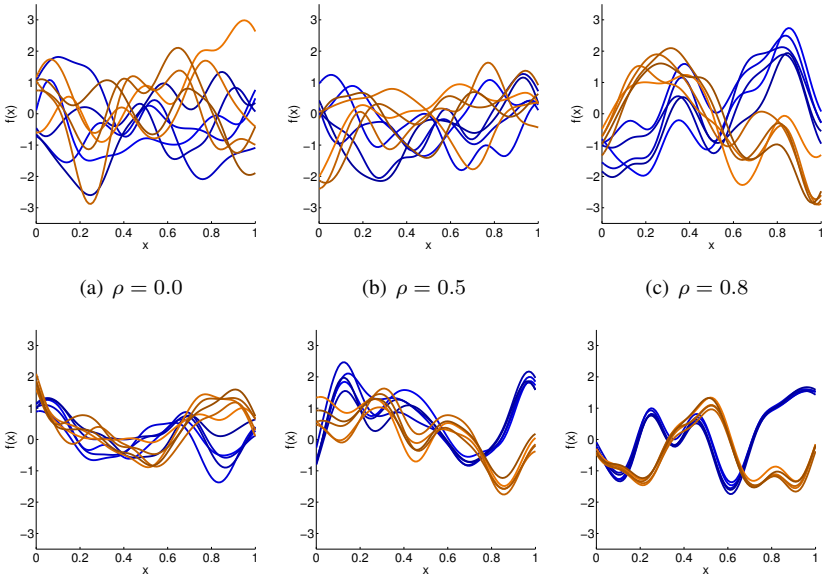
$$f^j(\boldsymbol{x}) = \tilde{f}^j(\boldsymbol{x}) + \alpha \cdot \bar{f}(\boldsymbol{x}) \ . \tag{3.27}$$

The resulting kernel function of the combined latent function for this model can be easily derived by using Lemma A.5 leading to:

$$\begin{aligned} K((j, \boldsymbol{x}), (j', \boldsymbol{x}')) &= \delta\left(j - j'\right) K^{\mathcal{X}}(\boldsymbol{x}, \boldsymbol{x}') + \alpha^2 K^{\mathcal{X}}(\boldsymbol{x}, \boldsymbol{x}') && (3.28) \\ &= \left(\delta\left(j - j'\right) + \alpha^2\right) \cdot K^{\mathcal{X}}(\boldsymbol{x}, \boldsymbol{x}') \ , && (3.29) \end{aligned}$$

which turns out to be a simplified version of Eq. (3.26) with additional assumptions, such as each task is equally related. A disadvantage is that the variance of the individual Gaussian processes increases with an increasing value of the task correlation parameter. Therefore, our approach uses the following parametrization of the combined kernel function which circumvents this problem:

$$K((j, \boldsymbol{x}), (j', \boldsymbol{x}')) = \begin{cases} K^{\mathcal{X}}(\boldsymbol{x}, \boldsymbol{x}') & \text{if } j = j' \\ \rho \, K^{\mathcal{X}}(\boldsymbol{x}, \boldsymbol{x}') & \text{otherwise} \end{cases} \ . \tag{3.30}$$



(a) $\rho = 0.0$ (b) $\rho = 0.5$ (c) $\rho = 0.8$

(d) $\rho = 0.9$ (e) $\rho = 0.95$ (f) $\rho = 0.99$

Figure 3.5: Two samples of a dependent Gaussian process prior with $P = 5$ latent functions, the Gaussian kernel as a base kernel function and a varying task correlation parameter ρ. The latent functions are differently shaded to show the correspondence to one of the displayed two samples. Note how the variance of the latent functions within one sample decreases with increasing ρ. The latent functions are independent if $\rho = 0$.

The hyperparameter ρ of the extended kernel function with $0 \leq \rho \leq 1$ controls the correlation of the tasks: $\rho = 0$ corresponds to the case of independent learning, whereas $\rho = 1$ assumes that the tasks are highly related. Let us have a look on some samples generated by this model depicted in Figure 3.5. We sampled two times from a joint Gaussian process equipped with the kernel function defined in Eq. (3.28). The two samples shaded differently generate $P = 5$ tasks represented by individual latent functions. We see that with an increasing task correlation parameter ρ, the expected difference between the tasks decreases.

The assumption of uniform task correlation ρ does not hold in general and especially not for heterogeneous learning environments with tasks of very different characteristics. For this reason, we use only one single support classification task that is automatically selected using the techniques described in Section 3.4.3 and Section 3.4.4.

3.4.2 Transfer Learning with Dependent Gaussian Processes

We now consider the case that two binary classification tasks are given: a support task with a large amount of training data $\mathcal{D}^{\mathcal{S}}$ and a target task with only few training examples $\mathcal{D}^{\tau}$. This setting is different from the scenario of multitask learning in which one wants to train classifiers for multiple binary classification tasks in combination. In our case, we do not want to improve the classifier for the support task. In the following, we consider transfer learning with the label regression approach as presented in Section 2.6.5. The same ideas can also be directly applied to approximate GP classification (Section 2.6.6). A comparison between transfer learning with label regression or GP classification is studied in Section 5.3.3.

The classification score of a traditional GP classifier not exploiting transfer learning is given by the following predictive mean of the label y_* of a new input example $\boldsymbol{x}_*$ (Section 2.6.5, Eq. (2.77)):

$$\mu_* = \boldsymbol{k}_*^T \left(\mathbf{K} + \sigma_\varepsilon^2 \cdot \mathbf{I}\right)^{-1} \boldsymbol{y} \ , \tag{3.31}$$

where we use the kernel vector $\boldsymbol{k}_* = K(\mathbf{X}, \boldsymbol{x}_*)$, the kernel matrix $\mathbf{K} = K(\mathbf{X}, \mathbf{X})$ of the training data $\mathbf{X}$, and the label vector $\boldsymbol{y} \in \{-1, 1\}^n$ as defined in Section 2.6.5. As elaborated in the previous section, transfer learning is done by using an extended kernel function. Therefore, in comparison to the single task GP model in Eq. (3.31), only the kernel function changes and the predictive mean of the target task label y_* can be calculated as follows:

$$\begin{aligned} \mu_* &= \boldsymbol{k}_*(\rho)^T (\mathbf{K}(\rho) + \sigma_\varepsilon^2 \cdot \mathbf{I})^{-1} \boldsymbol{y} \\ &= \begin{bmatrix} \boldsymbol{k}_{\tau *} \\ \rho \boldsymbol{k}_{s*} \end{bmatrix}^T \left(\begin{pmatrix} \mathbf{K}_{\tau\tau} & \rho \mathbf{K}_{\tau s} \\ \rho \mathbf{K}_{\tau s}^T & \mathbf{K}_{ss} \end{pmatrix} + \sigma_\varepsilon^2 \cdot \mathbf{I} \right)^{-1} \begin{bmatrix} \boldsymbol{y}_\tau \\ \boldsymbol{y}_s \end{bmatrix} \ , \end{aligned} \tag{3.32}$$

with $\boldsymbol{y}_\tau$ and $\boldsymbol{y}_s$ denoting the binary labels for the target and the support task, respectively. The matrix $\mathbf{K}_{\tau s}$ contains the pairwise kernel values of the target task and the support task. The same notational convention is used for $\mathbf{K}_{ss}, \mathbf{K}_{\tau\tau}$, $\boldsymbol{k}_{s*}$ and $\boldsymbol{k}_{\tau *}$. Note that we directly assumed that $\boldsymbol{x}_*$ is an example of the target task, *i.e.* $(\tau, \boldsymbol{x}_*)$ is given as an argument to the combined kernel function. The posterior variance for transfer learning with dependent GP is:

$$\begin{aligned} \sigma_*^2 &= K(\boldsymbol{x}_*, \boldsymbol{x}_*) - \boldsymbol{k}_*(\rho)^T \left(\mathbf{K}(\rho) + \sigma_\varepsilon^2 \cdot \mathbf{I}\right)^{-1} \boldsymbol{k}_*(\rho) + \sigma_\varepsilon^2 \\ &= K(\boldsymbol{x}_*, \boldsymbol{x}_*) - \end{aligned} \tag{3.33}$$

$$\begin{bmatrix} \boldsymbol{k}_{\tau *} \\ \rho \boldsymbol{k}_{s*} \end{bmatrix}^T \left(\begin{pmatrix} \mathbf{K}_{\tau\tau} & \rho \mathbf{K}_{\tau s} \\ \rho \mathbf{K}_{\tau s}^T & \mathbf{K}_{ss} \end{pmatrix} + \sigma_\varepsilon^2 \cdot \mathbf{I} \right)^{-1} \begin{bmatrix} \boldsymbol{k}_{\tau *} \\ \rho \boldsymbol{k}_{s*} \end{bmatrix} + \sigma_\varepsilon^2 \ . \quad (3.34)$$

Like the variance of single task learning given in Eq. (2.78), the variance of the multitask model does not depend on the labels. As proofed by Chai (2009), the posterior variance is a continuous and monotonically decreasing function in ρ and thus fulfills:

$$\sigma_*^2(\boldsymbol{x}_*, 1) \leq \sigma_*^2(\boldsymbol{x}_*, \rho) \leq \sigma_*^2(\boldsymbol{x}_*, 0) \ , \quad (3.35)$$

where the dependency of σ_*^2 on the task correlation ρ and the input example $\boldsymbol{x}_*$ is made explicit. This result is intuitive, because the assumption of higher correlation between the tasks should reduce the predicted uncertainty of the provided estimate.

3.4.2.1 Shared Background Category

In the context of image categorization, we are often confronted with one single background and multiple object categories (Torralba et al., 2007). Thus, all binary classification tasks share the background category. In this case the input sets $\mathbf{X}_s$ and $\mathbf{X}_\tau$ are not disjoint, which leads to an ill-conditioned kernel matrix $\mathbf{K}(\rho)$. We solve this problem by restricting the support training set only to examples of the object category. Therefore, the label vector $\boldsymbol{y}_s$ is always a vector of ones. Please note that due to our zero mean assumption of the GP prior, this leads to a valid classifier model. For independent learning ($\rho = 0$) it results in a one-class GP model for the support task, which is related to our one-class classification approach presented in Section 3.5.

3.4.2.2 Influence of the Task Correlation Parameter

As we have seen, an increasing value of the task correlation parameter ρ decreases the posterior variance. Let us now have a closer look on the influence of the task correlation parameter ρ on the resulting classification performance. Figure 3.6 shows the performance of the transfer learning approach applied to an image categorization application. We learned the object category *okapi* using 5 training examples and 30 examples of the support tasks *gerenuk*[3] and *chair*. The performance is measured using the area under the ROC curve (Section 5.1.2)

[3]The animal category *gerenuk* is also known as giraffe-necked antelope.

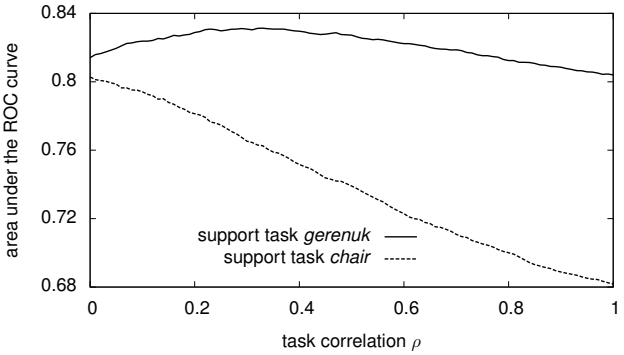


Figure 3.6: Performance of the GP-based transfer learning approach applied to the task *okapi* with 5 training examples and using two different support tasks and a varying task correlation parameter ρ. The additional bias at $\rho = 0$ is due to the transfer of the feature representation as described in Section 5.2.3.

and is plotted with respect to the task correlation parameter ρ. It can be seen that the optimal correlation parameter ρ using the support task *gerenuk* is around $\rho \approx 0.3$. Neither independent learning with $\rho = 0$, nor simply using all examples of the *gerenuk* category as training examples for the *okapi* category with $\rho = 1$ achieves a better performance. Therefore, finding a suitable value of the correlation parameter ρ is an important task, which is essential for the transfer learning algorithm to work in different scenarios.

In Section 3.4.3, we propose a method that is able to estimate an optimal value of ρ. Whereas using the support task *gerenuk* leads to an important improvement of the classification performance, utilizing the support task *chair* does not improve learning at all. This result is quite intuitive and highlights the importance of selecting an appropriate support task. In Section 3.4.4, we show how to pre-select a set of related categories using semantic similarities. Details about the previous experiment are given in Section 5.2.3.

3.4.3 Automatic Selection of Support Classes using Leave-One-Out

The optimization of the hyperparameter ρ and the selection of an appropriate support task can be handled as a combined model selection and a kernel hyperparameter optimization problem. In Section 2.6.10, we described how to optimize hyperparameters of the kernel function using the marginal likelihood derived from the GP framework. However, we show in Section 5.3.3 that the conditional likelihood $p(\boldsymbol{y}_\tau \mid \boldsymbol{y}_s, \mathbf{X}_s, \mathbf{X}_\tau^T)$ is not an appropriate model selection criterion in our transfer learning setting.

To solve this problem, we use leave-one-out estimates similar to Tommasi and Caputo (2009). In the context of Gaussian process regression, the posterior of the label of a training example $\boldsymbol{x}_i$ conditioned on all other training examples can be computed in closed form (Rasmussen and Williams, 2005, Section 5.4.2):

$$\log p(y \mid \mathbf{X}, \boldsymbol{y}_{-i}, \rho) = -\frac{1}{2}\log \tilde{\sigma}_i^2 - \frac{(y-\tilde{\mu}_i)^2}{2\tilde{\sigma}_i^2} - \frac{1}{2}\log 2\pi \ , \tag{3.36}$$

with $\tilde{\sigma}_i^2$ being the variance of the leave-one-out estimate $\tilde{\mu}_i$:

$$\tilde{\sigma}_i^2 = 1/\left(\mathbf{K}(\rho)^{-1}\right)_{ii} \quad \text{and} \quad \tilde{\mu}_i = y_i - \left(\mathbf{K}(\rho)^{-1}\boldsymbol{y}\right)_i \tilde{\sigma}_i^2 \ . \tag{3.37}$$

The estimates $\tilde{\mu}_i$ offer the possibility to use a wide range of model selection criteria, such as leave-one-out log predictive probability (Rasmussen and Williams, 2005) or squashed and weighted variants (Tommasi and Caputo, 2009). A common measure to assess the performance of a binary classification task is average precision (Section 5.1.2). Therefore, we calculate the average precision directly using the estimates $\tilde{\mu}_i$ and ground truth labels y_i. This decision is justified by experiments in Section 5.3.3, which compare average precision to multiple model selection criteria embedded in our approach.

We optimize the average precision with respect to ρ, which is a simple one-dimensional optimization, with golden section search (Kiefer, 1953) for each of the given support tasks. The task and corresponding value of ρ that yield the best average precision are chosen to build the final classifier according to Eq. (3.32).

3.4.4 Automatic Pre-Selection using Semantic Similarities

Selecting a support classification task among a large set of available tasks is itself a very difficult problem, and the selection method described above, might not be

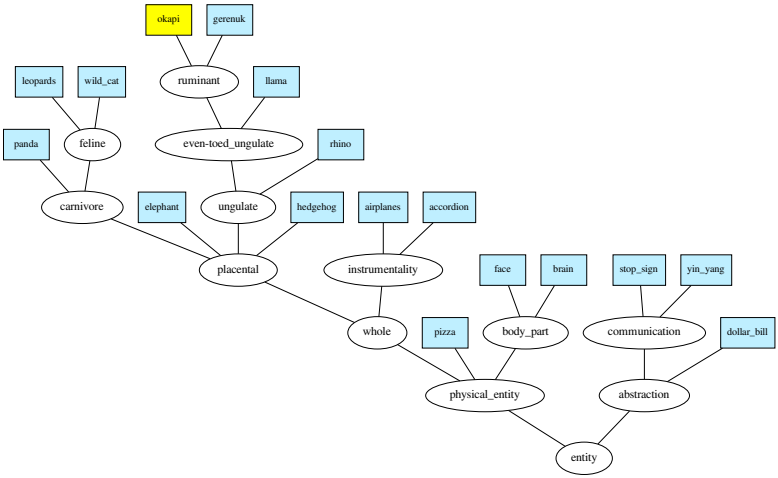


Figure 3.7: Reduced WordNet hierarchy of 17 categories from the Caltech-101 database. Rectangle boxes highlight object categories and ellipsoids represent superordinate terms, which are skipped if they only have one subordinate in the hierarchy. The exact mapping between category terms and WordNet senses is listed in Table B.1.

able to transfer beneficial information. A solution is the use of prior knowledge from other information sources to pre-select tasks which are likely to be related.

We optionally use the textual label of each object category together with WordNet, which is a hierarchical lexical database of the English language. The usefulness of this information source has been demonstrated recently in the context of attribute based knowledge transfer (Rohrbach et al., 2010b) and hierarchical classification (Marszalek and Schmid, 2007). Figure 3.7 shows a small part of the hierarchy derived from the "is-a" relationship included in WordNet. A common assumption is that semantically related object categories are also visual similar. Thus, the support task could be selected by semantic similarity measures such as the Resnik measure (Resnik, 1995) or simple inverse path lengths (Pedersen et al., 2004).

Whereas the similarity assumption might hold for certain areas, *e.g.* animal

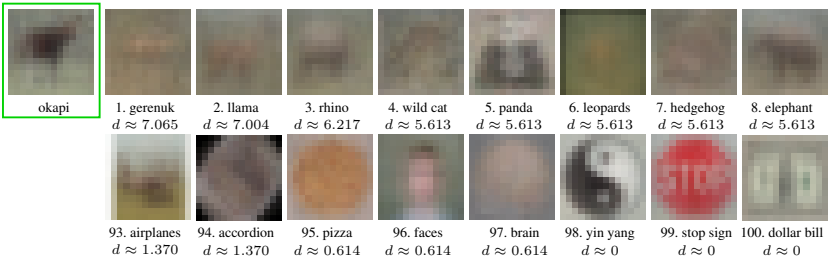


Figure 3.8: Ranking of support categories for the category *okapi* according to semantic similarities d computed using WordNet. The eight most similar and dissimilar categories are displayed as mean images using all images of the Caltech-101 database and inspired from Ponce et al. (2006).

hierarchies, it might not hold in all cases and prevents important knowledge transfer from other parts of the WordNet hierarchy with shared characteristics. However, one conclusion of the empirical analysis of Deselaers and Ferrari (2011) is that "visual similarity grows with semantic similarity". Therefore, we use WordNet in advance to leave-one-out selection and pre-select the J most related tasks among all available tasks based on their semantic similarity. For $J = 1$, WordNet selects the support task using the semantic of the category name and the leave-one-out method only optimizes the task correlation parameter ρ. If J equals the number of available support tasks, WordNet pre-selection does not influence transfer learning and the selection is based on visual similarity only. The importance of the combination of visual and semantic similarities for the selection is analyzed empirically in Section 5.3.3.

Figure 3.8 shows the ranking of Caltech-101 categories (Fei-Fei et al., 2006), as derived from the Resnik measure according to the relation to the category *okapi*, which follows the hierarchical structure given in Figure 3.7. We additionally display the mean appearance of each category using all corresponding images in the database. Taking a closer look to the mean images and the ranking position reveals that images of categories with a low semantic similarity value seem to be indeed more visually dissimilar compared to the first ranked object categories. Thus, on a very rough level of visual comparison, which is dominated by the global context, the WordNet pre-selection method is able to filter out a large part of non-related categories. Note that we use a more complex

feature set which includes an object part representation rather than holistic features. Therefore, we are able to exploit similarities between object parts, such as the appearance of animal legs, which are not captured by the visualization in Figure 3.8.

3.4.5 Required Computation Time

Let us analyze the computation time required for learning with our dependent GP framework. We assume that each support task has $\mathcal{O}(\tilde{n})$ training examples. To compute the leave-one-out estimates, we have to calculate the inverse combined kernel matrix, which requires $\mathcal{O}((n+\tilde{n})^3)$ dominated by the Cholesky factorization. Therefore, calculating the performance measures for each support task and for Q values of ρ (greedy optimization, $Q \approx 10$ in our experiments) can be done in a total runtime of $\mathcal{O}(Q \cdot J \cdot (n+\tilde{n})^3)$ steps. Learning the final label regression classifier does not require any additional calculation, because the Cholesky factorization can be directly used to compute the kernel coefficients $\boldsymbol{\alpha}$. After learning, $\mathcal{O}(n+\tilde{n})$ steps are needed to calculate the predictive mean and classify a new test example.

We have seen that the runtime highly depends on the number of support tasks and the number of support training examples. In our experiments in Section 5.3.3, we use 99 support tasks and each of the tasks has 230 training examples. The average runtime of learning and classification obtained was $71.44s$ on current hardware.

3.4.6 Related Work

In the following section, we highlight some connections to previous and related work. On the one hand, the aim is to show the differences in theoretical assumptions or applications areas. On the other hand, the underlying similarities between the approaches are important to gain further insight into limitations and possible extensions.

3.4.6.1 Joint Optimization of Hyperparameters

One of the first papers investigating knowledge transfer with GP is the work of Lawrence et al. (2004). They show that the joint optimization of hyperparameters using all tasks can be highly beneficial. In Section 2.6.10, we already described

their joint optimization method in the case of the one-vs-all GP classifier, which reduced to the following optimization problem:

$$\hat{\boldsymbol{\eta}}^{\mathrm{ML}} = \underset{\boldsymbol{\eta}}{\operatorname{argmin}} \ \frac{J}{2} \log\det\left(\tilde{\mathbf{K}}_{\boldsymbol{\eta}}\right) + \frac{1}{2}\mathrm{trace}\left(\tilde{\mathbf{K}}_{\boldsymbol{\eta}}^{-1} \sum_{j=1}^{J} \boldsymbol{y}^j \left(\boldsymbol{y}^j\right)^T\right) \quad . \quad (3.38)$$

In their multitask scenario, the involved label vectors $\boldsymbol{y}^j$ contain the binary labels of task j. An important property of their method is that it can only be applied if the same set of training points is given for each task. For visual object recognition, these situations arise if images are labeled with multiple objects, such as in the *PASCAL VOC* dataset (Everingham et al., 2010). In the following, we refer to this learning scenario as *multi-output learning* (Andriluka et al., 2007). A similar approach is the method of (Zhang and Yeung, 2009) which assumes a shared prior on hyperparameters.

3.4.6.2 Dependent Gaussian Processes for Multi-output Learning

As we have seen, dependent Gaussian processes model task dependencies in a more direct manner allowing to transfer information beyond shared kernel functions. Bonilla et al. (2008) use a dependent GP model to solve multi-output regression. The goal of multi-output regression is to fit functions to several sets of outputs corresponding to one single set of inputs, *i.e.* input examples are given with multiple output values. In such a setting, the combined kernel matrix $\mathbf{K}$ induced by the kernel function given in Eq. (3.26) can be written in an elegant manner using the Kronecker product $\otimes$ as $\mathbf{K} = \mathbf{K}^f \otimes \mathbf{K}^{\mathcal{X}}$. By exploiting some properties of the Kronecker product, efficient inference techniques are derived by Bonilla et al. (2008). In general, their approach is equivalent to a model in which a matrix normal distribution is used as a prior for the matrix $\mathbf{F} = \left[\boldsymbol{f}^1, \ldots, \boldsymbol{f}^J\right] \in \mathbb{R}^{n \times J}$. A matrix normal distribution with zero mean has two kinds of covariance matrices $\boldsymbol{\Omega} = \mathbb{E}\left(\mathbf{F}^T\mathbf{F}\right)$ and $\boldsymbol{\Sigma} = \mathbb{E}\left(\mathbf{F}\mathbf{F}^T\right)$. If the random matrix $\mathbf{F}$ is vectorized, it can be shown that the resulting random vector is normal distributed with the combined covariance matrix $\boldsymbol{\Omega} \otimes \boldsymbol{\Sigma}$. Therefore, the two covariance matrices of the matrix normal distribution can be directly identified with $\mathbf{K}^f$ and $\mathbf{K}^{\mathcal{X}}$ as used in the approach of Bonilla et al. (2008). An important application of their method are robot inverse dynamics problems as presented by Chai et al. (2008).

Schwaighofer et al. (2005) learn an overall non-parametric kernel matrix jointly with a set of regression tasks. Experiments are done with a collaborative filtering application in which people judge the quality of art images. The main idea is, if a person likes an image, a friend of him might like this image too. Learning the covariance matrix in a non-parametric fashion is also proposed by Yu et al. (2007), which apply a shared prior on each individual covariance matrix. They argue that the use of t-Processes in contrast to GP leads to a more robust heterogeneous multitask learning approach. Teh et al. (2005) solve multitask regression tasks by using linearly combined latent functions for each task leading to a particular combined GP prior. Their approach is a generalization of the method of Bonilla et al. (2008) to multiple base kernels and degenerated GP models. Andriluka et al. (2007) propose a special form of dependent covariance functions. They consider transfer learning for binary classification tasks and use the informative vector machine framework of Lawrence et al. (2004). A general framework for dependent GP using a graph-theoretical notation is presented by Yu and Chu (2008). Melkumyan and Ramos (2009) constructs combined kernel functions by convolution and Fourier analysis. The recent work of Pillonetto et al. (2010) presents online learning with dependent GP.

Parallel to our work, Cao et al. (2010) used the same framework for regression problems, such as WiFi localization. In contrast, they select the task correlation parameter ρ using the conditional likelihood. In our experiments in Section 5.3.3, we show that this model selection criterion is not appropriate for our classification setting with shared background categories.

3.4.6.3 Theoretical Studies and Transfer Learning with GP Latent Variable Models

Regression with dependent Gaussian processes is analyzed theoretically by Chai (2009). In his work, the generalization error is bounded by the error obtained by independent learning ($\rho = 0$) and learning by using all examples of the support task directly ($\rho = 1$). This allows utilizing known lower and upper bounds of single task GP regression.

The work of Urtasun et al. (2008) proposes to learn a latent feature space used across tasks. They extend the Gaussian process latent variable model introduced by Lawrence (2005) to multiple tasks and a shared latent space.

3.4.6.4 Relationship to Adapted Least-Squares SVM

The transfer learning approach of Tommasi and Caputo (2009) is based on least-squares SVM (LS-SVM) (Suykens et al., 2002) presented in Section 2.6.8.2. Let us in the following assume that we are given a binary classification target task and a support task. The idea of Tommasi and Caputo (2009) and originally of Orabona et al. (2009) is to modify the regularization term of the LS-SVM optimization problem (2.104), such that hyperplanes $\boldsymbol{w}^{(\tau)}$ of the target task with a large distance to the support task hyperplane $\boldsymbol{w}^{(s)}$ are penalized:

$$\begin{aligned} &\underset{\boldsymbol{w}^{(\tau)} \in \mathbb{R}^D, b \in \mathbb{R}, \boldsymbol{\xi} \in \mathbb{R}^n}{\text{minimize}} && \frac{1}{2} \left|\left| \boldsymbol{w}^{(\tau)} - \beta \boldsymbol{w}^{(s)} \right|\right|_{\mathcal{H}}^2 + \frac{C}{2} \sum_{i=1}^{n} \xi_i^2 \\ &\text{subject to} && \forall i: \; y_i = \left(\left\langle \boldsymbol{w}^{(\tau)}, \phi(\boldsymbol{x}_i) \right\rangle_{\mathcal{H}} + b \right) + \xi_i \;\; . \end{aligned} \tag{3.39}$$

where $(y_i, \boldsymbol{x}_i)$ is a training example of the target task. The parameter β is similar to our correlation parameter ρ and allows controlling the strength of transfer. Tommasi and Caputo (2009) states that the optimal solution of their Adapted LS-SVM method is given by a linear combination of the support hyperplane model and a new model learned using training examples of the target task:

$$\boldsymbol{w}^{(\tau)} = \beta \boldsymbol{w}^{(s)} + \sum_{i=1}^{n} \alpha_i^{(\tau)} \phi(\boldsymbol{x}_i) \;\; . \tag{3.40}$$

However, the coefficients $\alpha_i^{(\tau)}$ depend on $\boldsymbol{w}^{(s)}$ and it can be shown that they are given by:

$$\boldsymbol{\alpha}^{(\tau)} = \left(\mathbf{K}_{\tau\tau} + \frac{1}{C} \mathbf{I} \right)^{-1} \left(\boldsymbol{y}_\tau - \beta \cdot \mathbf{K}_{\tau s} \cdot \boldsymbol{\alpha}^{(s)} \right) \tag{3.41}$$

$$= \left(\mathbf{K}_{\tau\tau} + \frac{1}{C} \mathbf{I} \right)^{-1} \underbrace{\left(\boldsymbol{y}_\tau - \beta \cdot \mathbf{K}_{\tau s} \cdot \left(\mathbf{K}_{ss} + \frac{1}{C} \mathbf{I} \right)^{-1} \boldsymbol{y}_s \right)}_{\text{adapted label vector } \boldsymbol{y}'_\tau} \;\; . \tag{3.42}$$

This result offers some further insight. Instead of using the labels of the target examples directly, the Adapted LS-SVM approach applies the support task classifier on all target training examples to calculate an adapted label vector $\boldsymbol{y}'_\tau$ which is used for inference. Similar to the derivation in Section 2.6.8.2, we

assumed that the bias is incorporated in the kernel function and all details can be found in Section A.6. From a weight space view (Bishop, 2006, Section 6.4.1), optimization problem (3.39) uses a Gaussian prior on $\boldsymbol{w}^{(\tau)}$ with mean value $\beta \boldsymbol{w}^{(s)}$.

Finding an optimal value and an appropriate support task is done by leave-one-out estimates and a modified weighted error rate:

$$err_{\text{lsvm}} = \sum_{i=1}^{n} \omega_i \left(\frac{1}{1 + \exp\left(-10.0 \left(y_i \cdot \tilde{\mu}_i - 1\right)\right)} \right) \quad , \tag{3.43}$$

with $\tilde{\mu}_i$ being the leave-one-out estimate of example i as defined in Section 3.4.3 and the weights ω_i defined as:

$$\omega_i = \begin{cases} \frac{n}{2n_1} & \text{if } y_i = 1 \\ \frac{n}{2(n-n_1)} & \text{if } y_i = -1 \end{cases} \quad . \tag{3.44}$$

The number of positive examples is denoted by n_1. Tommasi and Caputo (2009) also assumes different noise variances of positive and negative examples. An extension of their approach is presented in Tommasi et al. (2010) and allows using multiple support tasks and kernels jointly. We compare our approach to the results of Adapted LS-SVM in Section 5.3.2.

3.4.6.5 Linear Classifier Combination

A simple idea for transfer learning is linear classifier combination. If we assume that target training examples and support training examples are not correlated, but a new test example $\boldsymbol{x}_*$ is correlated to both sets, we can simplify the prediction Eq. (3.32) significantly:

$$\mu_* = \boldsymbol{k}_*(\rho)^T (\mathbf{K}(\rho) + \sigma_\varepsilon^2 \mathbf{I})^{-1} \boldsymbol{y} \tag{3.45}$$

$$\stackrel{*}{=} \begin{bmatrix} \boldsymbol{k}_{\tau *} \\ \rho \boldsymbol{k}_{s *} \end{bmatrix}^T \left(\begin{pmatrix} \mathbf{K}_{\tau\tau} & \mathbf{0} \\ \mathbf{0} & \mathbf{K}_{ss} \end{pmatrix} + \sigma_\varepsilon^2 \mathbf{I} \right)^{-1} \begin{bmatrix} \boldsymbol{y}_\tau \\ \boldsymbol{y}_s \end{bmatrix} \tag{3.46}$$

$$= \boldsymbol{k}_{\tau *}^T \left(\mathbf{K}_{\tau\tau} + \sigma_\varepsilon^2 \mathbf{I} \right)^{-1} \boldsymbol{y}_\tau + \rho \cdot \boldsymbol{k}_{s*}^T \left(\mathbf{K}_{ss} + \sigma_\varepsilon^2 \mathbf{I} \right)^{-1} \boldsymbol{y}_s \tag{3.47}$$

$$= \mu_*^{(\tau)} + \rho \cdot \mu_*^{(s)} \quad . \tag{3.48}$$

Thus, the prediction is a linear combination of the result of the target task and the support task GP classifier applied independently. The dependent GP model

is a generalization and allows incorporating correlations between all training examples.

3.5 One-Class Classification

In previous sections, we presented transfer learning methods that can be used to boost the performance in the presence of only few training examples for a new target task. We now turn to the problem of one-class classification (OCC) introduced in Section 1.5 and propose several approaches to OCC with Gaussian process (GP) priors. Our research has been published in Kemmler et al. (2010) as a result of an intense and fruitful joint work with *Michael Kemmler*.

In OCC learning scenarios, we have to deal with a large set of examples from a single class (positive examples) and zero learning examples from a counter class (negative examples). In the following, we propose several approaches to OCC with Gaussian process (GP) priors. We further investigate the suitability of approximate inference methods of GP classification for OCC, such as Laplace approximation (Section 2.6.7) or expectation propagation (Minka, 2001).

The remainder of this section is structured as follows. Building on the review of machine learning with Gaussian process priors in Section 2.6, we present our approach to one-class classification with GP in Section 3.5.2 with implementation details provided in Section 3.5.3. Theoretical properties of our method are given in Section 3.5.4, and Section 3.5.5 compares the approach to previous work, such as support vector data description (Tax and Duin, 2004).

3.5.1 Informal Problem Statement

First of all, let us have a closer look on the aim of OCC and its requirements. In contrast to traditional binary classification, the training set provided consists only of examples of a single class and we assume without loss of generality that all training examples are labeled with $y = 1$. Tax (2001) refers to this class as the target class, however, we use the term *positive class* to highlight the conceptual difference to transfer learning.

The goal of one-class classification is to learn the boundaries of the positive class from the given training examples and to choose them such that the risk of classifying a negative example (outlier) as a positive example is minimized (Tax, 2001, p. 1). A suitable boundary is given by a level set of the

posterior distribution, *i.e.* all points which satisfy $p(y = 1 \mid \boldsymbol{x}) = \xi$ for a given value $\xi \in \mathbb{R}$. As already elaborated in Section 1.5, it is impossible to estimate a hard decision boundary without further assumptions. Due to this reason, the problem statement is often relaxed to finding a suitable soft decision classifier that models the posterior and the threshold ξ is not determined directly, but can be estimated using another method utilizing additional application-specific assumptions (Tax and Duin, 2002). In a large number of applications, such as image retrieval (Chen et al., 2001; Lai et al., 2002), it is even sufficient to estimate a function $\nu : \mathcal{X} \to \mathbb{R}$, such that the order of the posterior is retained, *i.e.* $\nu(\boldsymbol{x}_1) > \nu(\boldsymbol{x}_2)$ implies that $p(y_1 = 1 \mid \boldsymbol{x}_1) > p(y_2 = 1 \mid \boldsymbol{x}_2)$.

Generative methods (Section 2.1), which model the class-specific density $p(\boldsymbol{x} \mid y = 1)$, can be directly applied to OCC by using Bayes' law:

$$p(y = 1 \mid \boldsymbol{x}) = \frac{p(\boldsymbol{x} \mid y = 1) \cdot p(y = 1)}{p(\boldsymbol{x} \mid y = 1) \cdot p(y = 1) + p(\boldsymbol{x} \mid y = -1) \cdot p(y = -1)} \ ,$$

and assuming a flat (improper) outlier distribution $p(\boldsymbol{x} \mid y = -1) = \textit{const.}$ as done by Tax (2001, p. 16) leads to

$$p(y = 1 \mid \boldsymbol{x}) = \frac{p(\boldsymbol{x} \mid y = 1)}{p(\boldsymbol{x} \mid y = 1) + \beta} \ , \tag{3.49}$$

for a constant $\beta > 0$ independent of the input $\boldsymbol{x}$. Therefore, the density of the input data belonging to the positive class is connected to the posterior by a strictly monotonically increasing function. Density estimation techniques, such as the Parzen estimator or Gaussian mixture models (Bishop, 2006), directly provide a scoring function ν satisfying the above mentioned requirement of preserving the order. In the following section, we show how to utilize a discriminative technique, namely classification with Gaussian processes as introduced in Section 2.6, to solve OCC tasks.

3.5.2 One-Class Classification with Gaussian Processes

Classification approaches based on the GP framework (Section 2.6) are discriminative methods (Section 2.1) and provide a model for the posterior $p(y_* \mid \boldsymbol{x}_*, \mathcal{D})$ of a label y_* of a new example $\boldsymbol{x}_*$. In contrast to generative classification techniques, they are often not directly applicable to one-class classification, because most of them rely on minimizing the empirical error obtained from the training

data. Without negative examples such error estimates only minimize the number of false negatives, which simply lead to the trivial solution of setting the score $\nu(\boldsymbol{x})$ to a constant value for every input. For example, consider SVM classification as presented in Section 2.5. With the common formulation, there are an infinite number of hyperplanes consistent with the training data and with an arbitrary large margin. Thus, the optimization problem is not bounded and can not be applied to OCC.

The key idea of our approach is the powerful ability of Gaussian process priors to model a probability distribution of latent functions. In contrast to other supervised classification methods, the GP framework allows tackling the OCC problem directly. Utilizing a properly chosen Gaussian process prior enables us to derive useful membership scores for OCC. The essential property we would like to incorporate as prior knowledge is that the scoring function should decrease smoothly outside of a specific area. We simply use a GP prior with a mean value smaller than observed values of the latent function, which is for OCC easily achievable with a zero mean, because we only deal with positive class labels $y = 1$. This promotes latent functions with values gradually decreasing when being far away from observed points. If we additionally choose a smooth covariance function, such as the Gaussian kernel (Section 2.4.2), we obtain a Gaussian process model that allows us to sample functions satisfying the above mentioned essential property with high probability. Figure 3.9 shows a one-dimensional example of the above principle for GP regression with an OCC training set. The predictive probability $p(y_* = 1|\boldsymbol{x}_*, \mathcal{D})$ can be utilized and due to the fact that it is determined by its first and second order moments, we can also study whether the predictive mean and variance can serve as alternative membership scores. Their suitability is illustrated in Figure 3.9. The regression function obtained from the predictive mean decreases for inputs distant from the training data and can therefore be directly utilized as an OCC measure. This behavior is natural if we reconsider the formula of the predictive mean (Eq. (3.50)) when using a Gaussian kernel

$$\mu_* = \sum_{i=1}^{n} \alpha_i \cdot K^{\text{gauss}}(\boldsymbol{x}_i, \boldsymbol{x}_*) = \sum_{i=1}^{n} \alpha_i \cdot \exp\left(-\gamma \, ||\boldsymbol{x}_i - \boldsymbol{x}_*||^2\right) \quad . \tag{3.50}$$

Thus, for $||\boldsymbol{x}_i - \boldsymbol{x}_*|| \to \infty$ (for all examples $\boldsymbol{x}_i$) the predictive mean equals zero independent of the coefficients α_i. Returning to Figure 3.9, we observe that in contrast to the predictive mean, the predictive variance σ_*^2 is increasing at the boundaries, which suggests that the negative variance value can serve as

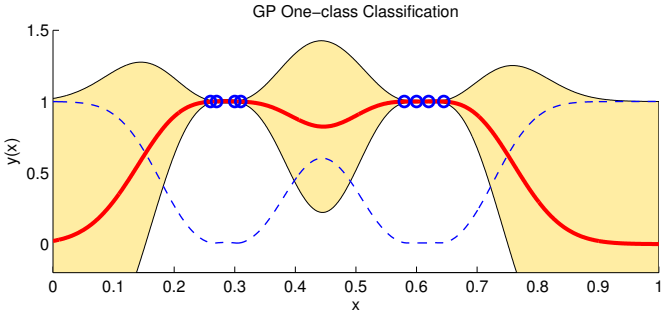
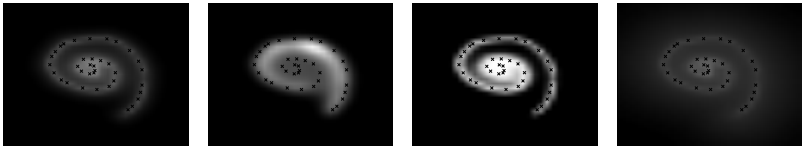


Figure 3.9: GP regression using a zero-mean GP prior in a one-dimensional OCC setting. The predictive distribution is visualized with its mean (red graph) and the corresponding confidence interval derived from the posterior variance. Training points are marked as circles. The dashed line shows the predictive variance as a function.

an alternative criterion for OCC. The latter concept is used in the context of clustering by Kim and Lee (2006). Additionally, Kapoor et al. (2010) propose the predictive mean divided by the standard deviation as a combined measure for describing the uncertainty of the estimation and applied this heuristic successfully in the field of active learning. All variants, which are summarized in Table 3.1, are available for GP regression and approximate GP classification with Laplace approximation (Section 2.6.7) or expectation propagation (Minka, 2001). The different membership scores produced by the proposed measures are visualized in Figure 3.10 using an artificial two-dimensional example. In spite of the simplicity of the approach, the ability of the GP framework to directly tackle OCC problems seems to have not been recognized previously. We show some

Table 3.1: Different measures derived from the predictive distribution which are all suitable for OCC membership scores $\nu(\boldsymbol{x}_*)$.

mean (M)	$\mu_* = \mathbb{E}(y_*\|\boldsymbol{x}_*, \mathcal{D})$	probability (P)	$p(y_* = 1\|\boldsymbol{x}_*, \mathcal{D})$
neg. variance (V)	$-\sigma_*^2 = -\sigma^2(y_*\|\boldsymbol{x}_*, \mathcal{D})$	heuristic (H)	$\mu_* \cdot \sigma_*^{-1}$



(a) GP-Reg-P (b) GP-Reg-M (c) GP-Reg-V (d) GP-Reg-H

Figure 3.10: One-class classification using GP regression (GP-Reg) and measures listed in Table 3.1. All measures model the complex distribution.

interesting connections to other approaches in Section 3.5.4 and Section 3.5.5.

3.5.3 Implementation Details

The implementation of GP for one-class classification is simple and straightforward in the regression case, especially for the posterior mean. First of all, the kernel matrix $\mathbf{K}$ has to be computed with an arbitrary kernel function such as the Gaussian kernel. The only thing which has to be done for training is to solve the linear equation system $\left(\mathbf{K} + \sigma_\varepsilon^2 \cdot \mathbf{I}\right) \boldsymbol{\alpha} = \mathbf{y}$ with $\boldsymbol{y}$ being an n-dimensional vector of ones. A solution can be found with the Cholesky decomposition, which involves $\mathcal{O}(n^3)$ operations. Afterward, the estimated posterior mean is $\boldsymbol{k}_*^T \boldsymbol{\alpha}$ which involves $\mathcal{O}(n)$ operations. Calculating the posterior variance is similar but involves $\mathcal{O}(n^2)$ operations during testing.

3.5.4 Connections and Other Perspectives

In the following, we present some connections of our approach to common techniques and we study our method from different perspectives. The derived relations offer to give further insights into underlying assumptions.

3.5.4.1 Predictive Mean Generalizes Gaussian Distributions and the Parzen Estimator

The relation to density estimation with the simple assumption of normally distributed input data becomes obvious by considering the case of a single training point $\boldsymbol{x}$ ($n = 1$) and the use of a Gaussian kernel with hyperparameter

$\gamma = \left(2\sigma^2_{\text{kernel}}\right)^{-1}$. These assumptions simplify the formula of the predictive mean as follows:

$$\mu_* = \left(\frac{1}{1+\sigma_\varepsilon^2}\right) \exp\left(-\frac{1}{2\sigma^2_{\text{kernel}}} ||\boldsymbol{x}_* - \boldsymbol{x}||^2\right) \quad . \tag{3.51}$$

If we assume noise-free observations ($\sigma_\varepsilon^2 = 0$), the right hand side of the equation is equivalent to an unnormalized normal distribution with mean value $\boldsymbol{x}$.

Another popular technique for density estimation is the *Parzen estimator* (Scott and Sain, 2004), also known as *kernel density estimation*. To obtain a density estimate at a new example $\boldsymbol{x}_*$, this non-parametric method sums up the similarity values of $\boldsymbol{x}_*$ with each of the training examples and multiplies it with a suitable normalization factor. If the similarity values are computed with a Gaussian kernel, this method is equivalent to smoothing the empirical data distribution with a Gaussian filter. Our OCC approach that uses the predictive mean derived from GP regression has a tight relationship to Parzen estimators. Let us assume noise-free observations ($\sigma_\varepsilon^2 = 0$) and no correlations between training examples ($\mathbf{K} = \mathbf{I}$), the regression mean (2.77) simplifies to

$$\mu_* = \boldsymbol{k}_*^T \mathbf{K}^{-1} \boldsymbol{y} \propto \frac{1}{n} \sum_{j=1}^{n} \sum_{i=1}^{n} \left(\mathbf{K}^{-1}\right)_{ij} \cdot K(\boldsymbol{x}_*, \boldsymbol{x}_i) = \frac{1}{n} \sum_{i=1}^{n} K(\boldsymbol{x}_*, \boldsymbol{x}_i) \tag{3.52}$$

where by construction we only have examples labeled as positive examples, *e.g.* $\boldsymbol{y} = (1, \ldots, 1)^T$. Therefore, one of our proposed OCC criteria can be seen as unnormalized Parzen density estimation using an additional scaling induced by the kernel matrix $\mathbf{K}$.

3.5.4.2 Feature Space Perspective of the Predictive Mean

Another insight offers a feature space perspective of the predictive mean of GP regression for OCC. Let $\boldsymbol{\Phi} = [\phi(\boldsymbol{x}_1), \ldots, \phi(\boldsymbol{x}_n]$ be the matrix containing all transformed training examples as columns. In the following, we make use of the relationship between the regularized kernel matrix $\mathbf{K}_{\text{reg}} = \mathbf{K} + \sigma_\varepsilon^2 \cdot \mathbf{I}$ and the regularized (sample) second moment matrix of the data in feature space $\mathbf{C}_{\text{reg}} = \frac{1}{n}\left(\boldsymbol{\Phi}\boldsymbol{\Phi}^T + \sigma_\varepsilon^2 \cdot \mathbf{I}\right)$ with $\sigma_\varepsilon^2 > 0$:

$$\boldsymbol{\Phi}\mathbf{K}_{\text{reg}}^{-1} = \frac{1}{n}\mathbf{C}_{\text{reg}}^{-1}\boldsymbol{\Phi}. \tag{3.53}$$

This result is proved in Section A.5 and originally from Haasdonk and Pękalska (2010, Section 3.2). We now turn back to the predictive mean of GP regression. Applying Eq. (3.53) allows writing the OCC inference equation as a scalar product of the mean $\boldsymbol{\mu}_{\boldsymbol{\Phi}}$ of all training data in feature space and the transformed new example $\boldsymbol{x}_*$:

$$\mu_* = \boldsymbol{k}_*^T \left(\mathbf{K} + \sigma_\varepsilon^2 \cdot \mathbf{I}\right)^{-1} \boldsymbol{y} = \phi(\boldsymbol{x}_*)^T \boldsymbol{\Phi} \mathbf{K}_{\text{reg}}^{-1} \boldsymbol{y} = \frac{1}{n} \phi(\boldsymbol{x}_*)^T \mathbf{C}_{\text{reg}}^{-1} \boldsymbol{\Phi} \boldsymbol{y} \quad (3.54)$$

$$= \phi(\boldsymbol{x}_*)^T \mathbf{C}_{\text{reg}}^{-1} \boldsymbol{\mu}_{\boldsymbol{\Phi}} \ . \quad (3.55)$$

Thus, our OCC approach using the predictive mean, indirectly measures the novelty of an example by comparing it to the mean in feature space by a normalized correlation.

3.5.4.3 Predictive Variance Models a Gaussian in Feature Space

A common way of describing a data distribution is to assume an underlying normal distribution. If we use a kernel-based approach, this assumption could be utilized in feature space rather than in input space. Using the result of Eq. (3.53) already given in the previous section, Haasdonk and Pękalska (2010) show the equivalence of the variance term in GP regression and a Mahalanobis distance in feature space. Let $\mathbf{C}_{\text{reg}}$ and $\mathbf{K}_{\text{reg}}$ be the regularized second moment and the kernel matrix as defined prior to Eq. (3.53). The regularized second moment matrix acts as a linear operator as follows:

$$\mathbf{C}_{\text{reg}} \phi(\boldsymbol{x}_*) = \frac{1}{n} \left(\boldsymbol{\Phi}\boldsymbol{\Phi}^T + \sigma_\varepsilon^2 \mathbf{I}\right) \phi(\boldsymbol{x}_*) \quad (3.56)$$

$$= \frac{1}{n} \left(\boldsymbol{\Phi}\boldsymbol{\Phi}^T \phi(\boldsymbol{x}_*) + \sigma_\varepsilon^2 \phi(\boldsymbol{x}_*)\right) \quad (3.57)$$

$$= \frac{1}{n} \left(\boldsymbol{\Phi}\boldsymbol{k}_* + \sigma_\varepsilon^2 \phi(\boldsymbol{x}_*)\right) \ . \quad (3.58)$$

Due to the fact that $\mathbf{C}_{\text{reg}}$ is invertible, we can multiply the above equation with $\phi(\boldsymbol{x}_*)^T \mathbf{C}_{\text{reg}}^{-1}$ and use Eq. (3.53), which leads to:

$$\phi(\boldsymbol{x}_*)^T \phi(\boldsymbol{x}_*) = \frac{1}{n} \left(\phi(\boldsymbol{x}_*)^T \mathbf{C}_{\text{reg}}^{-1} \boldsymbol{\Phi}\boldsymbol{k}_* + \sigma_\varepsilon^2 \phi(\boldsymbol{x}_*)^T \mathbf{C}_{\text{reg}}^{-1} \phi(\boldsymbol{x}_*)\right) \quad (3.59)$$

$$= \phi(\boldsymbol{x}_*)^T \boldsymbol{\Phi} \mathbf{K}_{\text{reg}}^{-1} \boldsymbol{k}_* + \frac{\sigma_\varepsilon^2}{n} \phi(\boldsymbol{x}_*)^T \mathbf{C}_{\text{reg}}^{-1} \phi(\boldsymbol{x}_*) \quad (3.60)$$

$$= \boldsymbol{k}_*^T \mathbf{K}_{\text{reg}}^{-1} \boldsymbol{k}_* + \frac{\sigma_\varepsilon^2}{n} \phi(\boldsymbol{x}_*)^T \mathbf{C}_{\text{reg}}^{-1} \phi(\boldsymbol{x}_*) \ . \tag{3.61}$$

Rearranging and using $K(\boldsymbol{x}_*, \boldsymbol{x}_*) = \phi(\boldsymbol{x}_*)^T \phi(\boldsymbol{x}_*)$ gives the relation of the predictive variance and a Mahalanobis distance in feature space:

$$\phi(\boldsymbol{x}_*)^T \mathbf{C}_{\text{reg}}^{-1} \phi(\boldsymbol{x}_*) = \frac{n}{\sigma_\varepsilon^2} \left(K(\boldsymbol{x}_*, \boldsymbol{x}_*) - \boldsymbol{k}_*^T \mathbf{K}_{\text{reg}}^{-1} \boldsymbol{k}_* \right) = \left(\frac{n}{\sigma_\varepsilon^2} \right) \sigma_*^2 \ . \tag{3.62}$$

Thus, the predictive variance of GP regression is proportional to the negative logarithm of the normal density in feature space with zero mean and a covariance matrix, which in this case equals to the second moment matrix, estimated from the training set:

$$\sigma_*^2 \propto -\log \mathcal{N}\left(\phi(\boldsymbol{x}_*) \,|\, \mathbf{0}, \mathbf{C}_{\text{reg}}\right) \ . \tag{3.63}$$

All multiplicative and additive constants are independent of the input example $\boldsymbol{x}_*$. If we use vectors $\tilde{\phi}(\boldsymbol{x})$ centered at the data mean, *i.e.*

$$\tilde{\phi}(\boldsymbol{x}) = \phi(\boldsymbol{x}) - \boldsymbol{\mu}_{\boldsymbol{\Phi}} = \phi(\boldsymbol{x}) - \frac{1}{n} \sum_{i=1}^{n} \phi(\boldsymbol{x}_i) \ , \tag{3.64}$$

the normal distribution involved in Eq. (3.63) does not have a zero mean but the correct mean in feature space. Centering in feature space can be done indirectly by modifying the kernel function.

3.5.5 Related Work

In the following section, we give a brief review of related work and highlight the similarities and differences to our approach. Tax and Juszczak (2002) mention the idea of estimating a normal distribution in feature space, which is connected to our use of the GP predictive variance as an OCC measure as shown in the previous section. Kim and Lee (2006) present a clustering approach that indirectly uses a GP prior and its predictive variance. Our new variants for OCC with GP priors include the idea of Kim and Lee (2006) as a special case and utilize it for OCC rather than clustering.

Proper density estimation with GP priors is studied by Adams et al. (2009). The idea is to model the data density $\tilde{p} : \mathcal{X} \to \mathbb{R}$ by squashing a latent function

$f : \mathcal{X} \to \mathbb{R}$ with a monotone function $\Phi : \mathcal{X} \to (0, 1)$:

$$\tilde{p}(\boldsymbol{x}) = \frac{\Phi(f(\boldsymbol{x})) \cdot \pi(\boldsymbol{x})}{Z_\pi^f} \quad , \tag{3.65}$$

where π is an arbitrary base probability measure and Z_π^f is a normalization factor given by:

$$Z_\pi^f = \int_{\mathcal{X}} \Phi(f(\boldsymbol{x}')) \cdot \pi(\boldsymbol{x}') \; d\boldsymbol{x}' \quad . \tag{3.66}$$

Assuming a GP prior for f implies a probability distribution for $\tilde{p}$, which is non-negative and normalized such that sampling from the distribution results in suitable density functions. Exact and closed-form inference is not possible with this model and Adams et al. (2009) presents a MCMC technique for approximate inference.

Nickisch and Rasmussen (2010) utilize the GP-LVM approach of Lawrence (2005) for density estimation. The key idea is to model the density in the latent space estimated by GP-LVM as a Parzen density or as a mixture of Dirac impulses corresponding to using the empirical density. Backprojecting this density to the original input space yields a flexible Gaussian mixture model with as many components as training examples. For parameter estimation they propose to use a leave-k-out procedure. The paper also offers a comparison and short review of several density estimation methods.

3.5.5.1 Support Vector Data Description

One-class classification with support vector data description (SVDD) (Tax and Duin, 2004) is one of the most popular OCC methods and is used for our experimental comparison in Chapter 5. The approach is based on the idea of enclosing the data with a hypersphere in feature space of minimal radius R. We can express this problem as a quadratic program similar to the soft margin version of SVM presented in Section 2.5.2, which utilizes additional slack variables ξ to account for outliers:

$$\begin{aligned} \underset{\boldsymbol{m} \in \mathcal{H}, R \in \mathbb{R}, \boldsymbol{\xi} \in \mathbb{R}^n}{\text{minimize}} \quad & R^2 + C \sum_{i=1}^{n} \xi_i \\ \text{subject to} \quad & \forall i : \; ||\phi(\boldsymbol{x}_i) - \boldsymbol{m}||_{\mathcal{H}}^2 \leq R^2 + \xi_i \;\text{ and }\; \xi_i \geq 0 \;\; . \end{aligned} \tag{3.67}$$

As in optimization problem (2.52) the parameter C controls the number of outliers, *e.g.* large values of C lead to a solution with only a small number of points outside the hypersphere. Schölkopf et al. (2001) use the parameter $0 \leq \nu \leq 1$ with $C = (n\nu)^{-1}$, which allows adjusting the fraction of expected outliers directly. Furthermore, the Euclidean distance in (3.67) can be generalized to Bregmanian distances (Crammer and Singer, 2003). If we turn the primal problem (3.67) into its corresponding dual optimization problem, we obtain a kernelized version of SVDD:

$$\begin{aligned} &\underset{\boldsymbol{\alpha} \in \mathbb{R}^n}{\text{maximize}} \quad \sum_{i=1}^{n} \alpha_i K_{ii} - \boldsymbol{\alpha}^T \mathbf{K} \boldsymbol{\alpha} \\ &\text{subject to} \quad \forall i = 1 \ldots n: \; 0 \leq \alpha_i \leq C \;\text{ and }\; \sum_{i=1}^{n} \alpha_i = 1 \;\; . \end{aligned} \tag{3.68}$$

The distance of a new example $\boldsymbol{x}_*$ to the center can be expressed by

$$||\phi(\boldsymbol{x}_*) - \boldsymbol{m}||_{\mathcal{H}}^2 = \boldsymbol{\alpha}^T \mathbf{K} \boldsymbol{\alpha} - 2\boldsymbol{k}_*^T \boldsymbol{\alpha} + K(\boldsymbol{x}_*, \boldsymbol{x}_*) \;\; , \tag{3.69}$$

and the radius is given in terms of $\boldsymbol{\alpha}$ by:

$$R^2 = \underset{i \in \mathcal{I}}{\text{argmax}} \;\; ||\phi(\boldsymbol{x}_i) - \boldsymbol{m}||_{\mathcal{H}}^2 \tag{3.70}$$

$$= \underset{i \in \mathcal{I}}{\text{argmax}} \;\; K_{ii} - 2 \sum_{j=1}^{n} \alpha_i K_{ij} \;\; , \tag{3.71}$$

where $\mathcal{I} = \{i \,|\, \alpha_i < C\}$ denotes the set of all inliers (Tax and Duin, 2004).

There exists a tight relationship to 1-SVM as presented by Schölkopf et al. (2001). For kernels with $K(\boldsymbol{x}, \boldsymbol{x}) = \textit{const.}$ for all inputs $\boldsymbol{x}$, the SVDD problem is equal to finding the hyperplane which separates the data from the origin with largest margin (Schölkopf et al., 2001). Instead of using the hinge loss, we could also use a quadratic loss, which leads to least-squares SVM (Suykens et al., 2002). In Section 2.6.8.2 we also illustrated the relationship between GP regression and least-squares SVM. Due to this reason, our extension of GP techniques to one-class classification problems corresponds to the work of Choi (2009) in this specific case.

3.5.5.2 Theoretical Considerations

Consistency is an important property of an estimator, because it ensures that the estimated value converges to the correct value with high probability for $n \to \infty$. Vert and Vert (2006) show the consistency of a multitude of estimators with a similar underlying optimization problem as used in the one-class SVM formulation of Schölkopf et al. (2001) and which utilizes a normalized Gaussian kernel:

$$\underset{f \in \mathcal{H}_\sigma}{\text{minimize}} \quad \frac{1}{n} \sum_{i=1}^{n} \Phi(y_i \cdot f(\boldsymbol{x}_i)) + \lambda \, ||f||^2_{\mathcal{H}_\sigma} \quad . \tag{3.72}$$

where Φ is an arbitrary convex loss function. The feature space induced by a normalized Gaussian kernel with hyperparameter σ^2 (Vert and Vert, 2006, p. 2) is denoted by $\mathcal{H}_\sigma$ and plays an important role in the results of Vert and Vert (2006). The consistency is only ensured if σ is decreasing for an increasing number of training examples. This condition is analogous to the consistency requirements of the Parzen estimator (Scott and Sain, 2004), where σ^2 is often referred to as bandwidth parameter. Optimization problem (3.72) is not only a generalization of the one-class SVM problem of Schölkopf et al. (2001) but also a generalization of our one-class GP approach when using the predictive mean. This relationship can be seen by setting $\Phi(z) = (1 - z)^2$ and considering the alternative formulations of GP regression in Section 2.6.8. In our experiments in Section 5.5, we observe and analyze some important consequences related to the requirement of a decreasing hyperparameter of the kernel.

Chapter 4

Visual Categorization

The previous chapter concentrated on new machine learning approaches for transfer learning and one-class classification. Applying these approaches to visual recognition tasks requires extracting appropriate and discriminative features of all given images. Visual categorization refers to tagging images or videos with a single category label and is different to *object detection* or *semantic segmentation*, which correspond to the tasks of finding objects and estimate bounding boxes or labeling each image pixel. Therefore, questions like "Is there an okapi in this image?" are answered rather than "Where is an okapi in this image?". A related term is *generic object recognition*, which can be seen as a synonym for image categorization but emphasizes the usage of object categories rather than general semantic concepts, *e.g.* "outdoor", "open country" or "mountains" (Bosch et al., 2008).

The following chapter focuses on computer vision techniques used in this thesis for image categorization, such as feature extraction methods and image-based kernel functions. There is a large body of literature related to those topics (the keywords "*bag visual words classification*" give more than 45.000 search results in Google scholar) and in this thesis we only consider the main algorithms that allow us to evaluate our methods presented in Section 3. Textbooks covering visual categorization, as done in current research, are Pinz (2005) and Szeliski (2011, Section 14.4).

Figure 4.1 provides a visual outline of the current chapter. In Section 4.1, we briefly present local features, such as the standard *scale invariant feature*

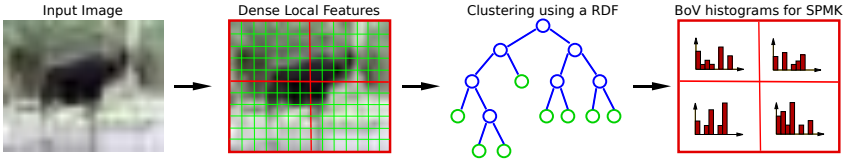


Figure 4.1: Local features are computed on a grid without using interest point detectors. All local features are clustered with a randomized decision forest and used to compute bag-of-visual-words (BoV) histograms for each cell. Only one level of the pyramid is shown in this figure.

transform (SIFT) approach of Lowe (2004) and its generalizations to color images as presented by van de Sande et al. (2010). The *bag of visual words (BoV)* principle used in our experiments and our computer vision applications is described in Section 4.2. Kernel functions allow incorporating application-specific prior knowledge into the learning process. Section 4.3 reviews image-based kernel functions, *i.e.* methods that calculate the similarity between images and satisfy the Mercer condition (Theorem (2.3)).

4.1 Local Features

Local features are currently one of the key ingredients of high-level computer vision approaches like visual object recognition (van de Sande et al., 2010) and 3d reconstruction (Agarwal et al., 2009; Snavely et al., 2006). Their application in current computer vision research is omnipresent and it seems to be hard finding a paper not using local features as one of the fundamental parts. Local features are often motivated by part decompositions of objects (Fergus et al., 2003), which can not be handled by global or holistic features (Torralba, 2003). The aim of local features is to represent an image with a set of vectors $\mathcal{L} = \{(\boldsymbol{p}^{(k)}, \boldsymbol{l}^{(k)})\}_{k=1}^{W}$ that is robust with respect to different transformations and distortions of the image, such as rotations and illumination changes in the scene. The term "local feature" always refers to two aspects: a feature detector and a local descriptor. The detector tries to find locations (keypoints) $\boldsymbol{p}^{(k)}$ in the image that are easy to relocate in subsequent views of the scene such as corners. In contrast, the

descriptor calculates a feature vector $\boldsymbol{l}^{(k)} \in \mathcal{U}$ of fixed dimensionality for each position $\boldsymbol{p}^{(k)}$.

4.1.1 Dense Sampling

In the following, we restrict our presentation to local feature descriptors rather than detectors (Mikolajczyk et al., 2005). In our visual categorization experiments, we calculate local features on an image grid and on several fixed scales. As suggested by van de Sande et al. (2010) and realized in their software, we use a triangular grid[1], which has the advantage that sampling is much more efficient, *i.e.* fewer sample points are needed to reconstruct a band-limited continuous signal (Dudgeon and Mersereau, 1984, Section 1.4.3). This method referred to as *dense sampling* often achieves better categorization results compared to representations based on estimated keypoints (Nowak et al., 2006). In our experiments, we use the scales $\sigma \in \{1.6, 3.2\}$ and a spacing of 10 pixels between each interest point.

4.1.2 SIFT Descriptor

The visual appearance of objects in images is often dominated by object boundaries or a specific texture with characteristic edges. Therefore, the main idea of the SIFT descriptor is to compute a histogram of gradient orientations. An important property of the gradient orientation is its invariance to global illumination changes. Let $\mathfrak{g} : \mathbb{R}^2 \rightarrow \mathbb{R}$ be a gray-value image function and $u : \mathbb{R} \rightarrow \mathbb{R}$ be a one-dimensional function that maps all gray-values $g = \mathfrak{g}(\boldsymbol{p})$ according to a global illumination change. By using the chain rule, we derive the following for the normalized gradient direction:

$$\frac{\{\nabla u(\mathfrak{g})\}(\boldsymbol{p})}{||\{\nabla u(\mathfrak{g})\}(\boldsymbol{p})||} = \left|\frac{du}{dg}(\mathfrak{g}(\boldsymbol{p}))\right|^{-1} \left(\frac{du}{dg}(\mathfrak{g}(\boldsymbol{p}))\right) \cdot \frac{\nabla \mathfrak{g}(\boldsymbol{p})}{||\nabla \mathfrak{g}(\boldsymbol{p})||} \quad . \tag{4.1}$$

Thus, the gradient orientation is not affected by strictly monotonically increasing global illumination changes, *i.e.* maps $u(\cdot)$ which do not change the order of gray-values. Note that the SIFT algorithm computes gradients not directly in the image, but in a smoothed image $\{\mathfrak{h}_\sigma * \mathfrak{g}\}(\boldsymbol{p})$ with $\mathfrak{h}_\sigma$ being the Gaussian

[1] The triangular grid is sometimes called hexagonal sampling (Dudgeon and Mersereau, 1984) or honeycomb structure (van de Sande et al., 2010).

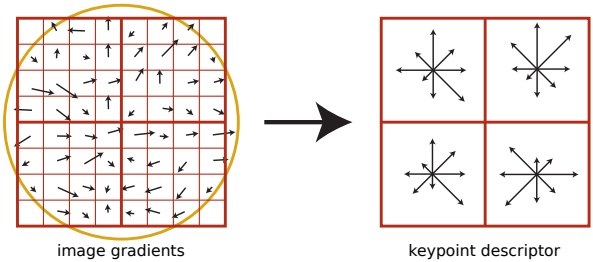


Figure 4.2: Calculation of the SIFT descriptor: (1) image gradients are calculated in a pixel neighborhood and weighted with an isotropic Gaussian mask; (2) an orientation histogram is computed for each block. Note that this visualization is simplified and uses a 8×8 rather than a 16×16 sample array (Lowe, 2004).

filter with scale σ. The scale is estimated by the detector or manually predefined, if scale-invariance is not important (Section 4.1.1). Therefore, the illumination invariance only holds approximately for small scales σ.

The standard SIFT descriptor of Lowe (2004) calculates the gradient strength and orientation in a neighborhood around a position with a 16×16 sample array. The size of the neighborhood depends on σ. Gradient orientations are quantized into 8 different directions and 16 histograms are build by grouping cells of 4×4 samples, which results in a $S = 128$ dimensional feature vector. The contribution to histogram cells is weighted with the gradient magnitude and an isotropic Gaussian centered at the feature position. Weighting with the gradient magnitude can be justified by an error analysis of the gradient orientation computation:

$$\theta_{\mathfrak{g}} = \arctan\left(\frac{\mathfrak{g}_y}{\mathfrak{g}_x}\right), \tag{4.2}$$

with $\mathfrak{g}_x$ and $\mathfrak{g}_y$ being the image gradients, *i.e.* $(\mathfrak{g}_x, \mathfrak{g}_y)^T = \nabla\mathfrak{g}(\boldsymbol{p})$. If the gradients are disturbed by $\triangle_x$ and $\triangle_y$, we can analyze the resulting error for $\theta_{\mathfrak{g}}$ by first-order Taylor approximation:

$$\left|\arctan\left(\frac{\mathfrak{g}_y + \triangle_y}{\mathfrak{g}_x + \triangle_x}\right) - \arctan\left(\frac{\mathfrak{g}_y}{\mathfrak{g}_x}\right)\right| \approx \frac{1}{||\nabla\mathfrak{g}||^2} \left|\mathfrak{g}_x \cdot \triangle_y - \mathfrak{g}_y \cdot \triangle_x\right| \quad . \tag{4.3}$$

Thus, the weighting can be seen as accounting partially[2] for the expected error made in calculating the orientation.

Grouping the orientations offers some robustness to small distortions. A rotational invariance can be achieved by storing orientation histograms relative to the main orientation of the local feature. The work of Zhang et al. (2007) shows that using a large degree of invariance is not beneficial for image categorization applications. For example, some objects have a main orientation in images, *i.e.* cars displayed upside down are unlikely. Figure 4.2 depicts the main steps involved. Our review of the SIFT algorithm omits some implementation details. A complete algorithm description can be found in the original paper of Lowe (2004) and an evaluation of different local descriptors is given by Mikolajczyk and Schmid (2005).

4.1.3 Local Color Features

The original SIFT descriptor is designed for gray-value images. However, visual object recognition benefits from additional color cues. For example, recognizing the category *fire truck* is a rather difficult task when considering gray-value images, but easier with access to color images and features. There is a huge variety of methods available that compute local color features. Building upon the early work of van de Weijer and Schmid (2006), van de Sande et al. (2010) give a comprehensive overview and evaluation in the context of image categorization and video retrieval. In our work, we use two different local features, rgSIFT and OpponentSIFT that achieved the best results. Whereas rgSIFT simply computes SIFT descriptors on all three normalized RGB channels and concatenates them, the OpponentSIFT approach uses the opponent color space as proposed by van de Weijer et al. (2005):

$$\begin{bmatrix} o_1 \\ o_2 \\ o_3 \end{bmatrix} = \begin{bmatrix} R-G \\ R+G-2B \\ R+G+B \end{bmatrix} = \begin{bmatrix} 1 & -1 & 0 \\ 1 & 1 & -2 \\ 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix} . \tag{4.4}$$

The components of the color space are denoted by o_1, o_2, o_3 and we skipped the additional multiplicative factors given by van de Sande et al. (2010) as they are

[2]Note that instead of using the quadratic gradient magnitude as suggested by Eq. (4.3), Lowe (2004) uses $||\nabla \mathfrak{g}||$ for weighting.

not relevant for SIFT descriptor calculation. The first two channels o_1 and o_2 are invariant with respect to a global constant added to the RGB vector. Note that o_3 is the non-weighted gray-value. For our experiments in Chapter 5, we utilize the software provided by van de Sande et al. (2010).

4.1.4 Local Range Features

Information about the real 3d structure of an object is normally lost during acquisition with standard 2d color cameras, but depth information can give important cues for object recognition tasks. In Section 5.8, we consider the combination of a standard 2d CCD camera and a *time-of-flight (ToF)* camera, which provides depth images in realtime. In this section, we briefly review local features working on range images, which were originally proposed by Hetzel et al. (2001) for specific object recognition and utilized by Hegazy (2009) and Rodner et al. (2010) for recognition of objects on a category level. All features are computed for each pixel and quantized to build local histograms.

Pixel Depth A local histogram of depth values is the simplest available feature. All values are naturally invariant with respect to image plane translations and rotations and provide a rough representation of the object shape. As pointed out by Hetzel et al. (2001), depth histograms can be misleading in scenarios with a large amount of background clutter and the presence of multiple objects.

Surface Normals A histogram of surface normals represented as a pair of two angles (ϕ, θ) in sphere coordinates provides a first-order statistic of the local object shape. Surface normals can be easily derived by approximating the gradient with finite differences. The use of surface normals for recognition tasks is also studied in our work on place recognition and rough self-localization of a robot (Kemmler et al., 2009), which is not part of this thesis.

Curvature and Shape Index Representing local curvature in the depth map can be done by using the shape index introduced in Koenderink and van Doorn (1992) and applied in Hetzel et al. (2001):

$$S(\boldsymbol{p}) = \frac{1}{2} - \frac{1}{\pi} \arctan \left(\frac{c_{\max}(\boldsymbol{p}) + c_{\min}(\boldsymbol{p})}{c_{\max}(\boldsymbol{p}) - c_{\min}(\boldsymbol{p})} \right), \tag{4.5}$$

where $c_{\max}(\boldsymbol{p})$ and $c_{\min}(\boldsymbol{p})$ are the principle curvatures computed at a point $\boldsymbol{p}$. The principal curvatures are proportional to the eigenvalues of the structure matrix (Lowe, 2004). Therefore, the second derivatives have to be computed in order to calculate the shape index.

4.2 Bag of Visual Words

The methods presented in the previous section allow us to compute a local feature representation $\mathcal{L}$ of an image. Subsequent feature computation steps completely rely on this representation without using any further information extracted from the image. There are two alternatives how to perform learning with such representations: (1) calculate a feature vector of fixed dimensionality or (2) use kernel functions two compare two local feature sets. In the following, we concentrate on how to turn a set of local features in a feature vector of fixed dimensionality with the bag of visual words (BoV)[3] approach (Dance et al., 2004; Willamowski et al., 2004). The second principle is explained in Section 4.3 and heavily relies on the BoV principle.

4.2.1 Images as Collections of Visual Words

The BoV approach can be introduced intuitively as orderless and plain object part representations. For example, if we know the image contains two eyes, four legs and several parts of striped fur, it is likely to contain a zebra or an okapi. This example translates to local feature representations as follows. Let us assume that each training image contains a single object and that local features capture object parts. If we consider the set of all local features $\bigcup_{i=1}^{n} \mathcal{L}^{(i)} = \{\boldsymbol{l}^{(k,i)}\}_{i=1,k=1}^{n,W}$ obtained from every training image i, typical object parts would cluster together in the high-dimensional input space $\mathcal{U}$, ($\mathbb{R}^{128}$ for SIFT features). If we additionally know the clusters and count the number of local features belonging to each cluster for each image, we arrive at the same type of features as used in our initial example.

The bag of visual words approach can be divided in two steps: (1) quantization of local features and (2) calculation of histograms for each image. Let us formulate these steps in a more detailed manner. In the first step, a suitable

[3] Alternative names for bag of visual words are: bag of features (BoF) (Lazebnik et al., 2006a), bag of keypoints (Willamowski et al., 2004), bag of words (BoW) (Niebles et al., 2008).

quantization of the set $\mathcal{U}$ has to be defined, which is mostly done by clustering algorithms or the random decision forest method reviewed in Section 4.2.2. A quantization (clustering or codebook) is a finite decomposition of the set $\mathcal{U}$, *e.g.* a function $q : \mathcal{U} \to \{1, \ldots, n_q\}$ which returns the corresponding cluster index of an element of $\mathcal{U}$. Equivalent terms for clusters are cells or codebook elements.

Building a histogram $\boldsymbol{h}^{(i)} = (h_1^{(i)}, \ldots, h_{n_q}^{(i)})^T$ for each image i is straightforward but care has to be taken of several important normalization aspects. If we consider the example given in the beginning of this section, we would use a histogram of absolute counts $c_j^{(i)}$:

$$\tilde{h}_j^{(i)} = c_j^{(i)} \stackrel{\text{def}}{=} \left| \left\{ k \,|\, q\left(\boldsymbol{l}^{(k,i)}\right) = j \right\} \right| \quad . \tag{4.6}$$

However, absolute counts are not invariant with respect to the number of local features W in an image. Due to this reason, relative counts seem to be more suitable:

$$h_j^{(i)} = c_j^{(i)} \cdot \left(\sum_{j'=1}^{n_q} c_{j'}^{(i)} \right)^{-1} \quad . \tag{4.7}$$

A third variant uses an additional binarization method applied to the histograms with thresholds estimated by mutual information (Dorkó and Schmid, 2003; Vidal-Naquet and Ullman, 2003; Nowak et al., 2006; Ullman et al., 2002; Epshtein and Ullman, 2006, 2005) or simply set to zero. Additional modifications include: smoothing histograms (Deselaers et al., 2005), applying topic models (Monay et al., 2005; Bosch et al., 2008) and its randomized extensions (Rodner and Denzler, 2009b) to build compact BoV representations. Marszalek and Schmid (2006) incorporate shape masks given as additional training information and used to perform a spatial weighting of local features. The work of Ommer and Buhmann (2007) uses the BoV histograms of ensembles of neighbored local features and combines their information and dependencies in a graphical model.

4.2.2 Supervised Clustering

Calculating BoV histograms as given in Eq. (4.7) is done both in training as in testing for each image. One important step during learning is to estimate a quantization of local features from all training examples. Common approaches apply standard clustering techniques, such as k-Means (Dance et al., 2004),

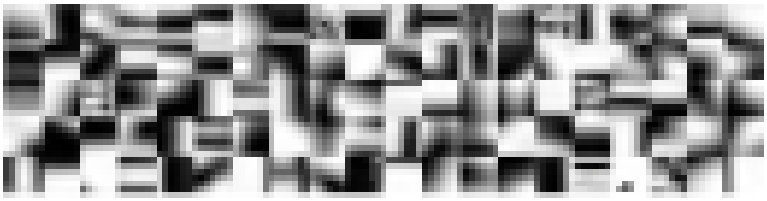


Figure 4.3: Example of a bag of visual words codebook computed by online k-Means (Nowak et al., 2006): each codebook element is represented by the mean local gray-value patch computed using all local features of the training images belonging to the corresponding cluster.

online k-Means (Nowak et al., 2006), Gaussian mixture models (Csurka and Perronnin, 2008) or density based clustering (Jurie and Triggs, 2005). The main disadvantage of these methods is that they do not exploit the given labels. Clustering is done in a completely unsupervised manner, although we would like to have clusters which are mainly formed by local features from a single category, i.e. clusters corresponding to typical category-specific object parts. Figure 4.3 shows a codebook obtained using online k-Means. It can be seen that a large part of the codebook contains elements related to different gradient orientations or other local image statistics. Specific object parts are not directly captured by this unsupervised clustering method.

The method of Moosmann et al. (2006b) allows building a discriminative clustering by using random decision forests. In Section 2.3.1, we already presented decision trees and mentioned their ability to partition the input space in a supervised manner. In a first step, Moosmann et al. (2006b) learn a random decision forest using all local features from all training images and their corresponding labels. Building BoV histograms is done afterward by considering the number of local features present in each leaf node θ_j normalized using the total number of local features R_i computed for image i:

$$h_j^{(i)} = \left| \left\{ \mathbf{f}^{(k,i)} \,|\, a(\mathbf{f}^{(k,i)}) = \theta_j \right\} \right| \cdot R_i^{-1} \,. \tag{4.8}$$

Note that this definition is equivalent to leaf probabilities defined in Section 3.2.2 when restricting the calculation to single images. The learning process of the decision tree (Section 2.3.1.1) guarantees that the leaf nodes are approximately homogeneous. Thus, a high histogram value can be an important clue about the

presence of a category in the image. The size of the codebook can be changed to a fixed value by pruning the random decision forest after the learning until the required number of leaf nodes is reached.

Despite being able to derive discriminative clusters, another important advantage is the short computational time needed to create a codebook. Whereas common algorithms, such as k-Means or Gaussian mixture models, have a quadratic runtime and are only tractable with previous subsampling (Hörster et al., 2007; Lazebnik et al., 2006a) or feature reduction (Wang et al., 2006), random decision forests can be quickly build and accelerated by increasing the degree of randomization (Section 2.3.2). An additional advantage of this method, is the availability of a local classifier, which can be used for building saliency maps (Moosmann et al., 2006a), which are probability maps showing the rough location of the object. The underlying ideas of the semantic segmentation approach of Shotton et al. (2008) are similar and also use a random decision forest as a local classifier and a clustering method in conjunction. Note that our published paper on semantic segmentation (Fröhlich et al., 2010) shows the usefulness of random decision forests and local features beyond image categorization for the task of facade detection.

4.3 Image-based Kernel Functions

A common way to use BoV histograms for image categorization is to use them as feature vectors and apply a linear or kernel classifier with a standard kernel, such as the Gaussian kernel (Section 2.4). The disadvantages of this procedure are: (1) lack of information about the underlying structure and density of the set of local features; (2) no location information is incorporated and images are represented as collections of unordered parts.

Coping with these issues can be done with image-based kernel functions, which are partly reviewed in the following section. Instead of extracting features of an image and applying a standard kernel function, image-based kernel functions can be seen as defining a kernel function directly on images rather than on plain vectors.

4.3.1 Pyramid Matching Kernel

The *pyramid match kernel (PMK)* of Grauman and Darrell (2007)[4] allows comparing two sets of multi-dimensional vectors. For image categorization the sets are local descriptors of two images, *i.e.* $\mathcal{L}^{(i)}$ and $\mathcal{L}^{(i')} \in \mathcal{P}(\mathcal{U})$. The PMK algorithm can be introduced as an approximation of the error of an optimal matching of both sets:

$$err_\pi = \min_{\text{matching } \pi} \sum_{k=1}^{W_i} \left|\left| \boldsymbol{l}^{(k,i)} - \boldsymbol{l}^{(\pi(k),i')} \right|\right|_1 \ , \tag{4.9}$$

where $\boldsymbol{l}^{(\cdot,i)}$ and $\boldsymbol{l}^{(\cdot,j)}$ are elements of $\mathcal{L}^{(i)}$ and $\mathcal{L}^{(j)}$, respectively, $||\cdot||_1$ is the L_1-norm and π is an injective map. Computing the matching error and solving the matching problem requires a cubic runtime $\mathcal{O}(W^3)$ in the number W of local descriptors using the Hungarian method (Kuhn, 2010). Thus, in order to develop an efficient method to compare two sets using their matching error, an approximation is needed.

The main idea of the PMK framework of Grauman and Darrell (2007) is to utilize L quantizations $q^{(\ell)} : \mathcal{U} \to \{1, \ldots, n_q^{(\ell)}\}$ of the feature space $\mathcal{U}$ with increasing resolution, *i.e.* $n_q^{(0)} < \ldots < n_q^{(L-1)}$ and the restriction that every cell of the quantization at level ℓ is a subset of one of the cells at level $\ell + 1$. Note that in the paper of Lazebnik et al. (2006a), the quantization of level zero has the highest resolution, *i.e.* largest number of cells. In contrast, we stick to the conventions used in Grauman and Darrell (2007), which define level zero to have the smallest number of cells. Grauman and Darrell (2007) divide each dimension in 2^ℓ equivalent ranges to construct the quantization $q^{(\ell)}$. An extension using data-dependent quantizations is provided by Grauman et al. (2006). With the set of quantizations, a notion of approximate matching can be defined. Two elements of $\mathcal{U}$ are said to approximately match at level ℓ if they belong to the same cell of the partitioning. The pyramid match kernel is the approximate matching similarity defined as:

$$K^{\text{PMK}}(\mathcal{L}^{(i)}, \mathcal{L}^{(i')}) = \sum_{\ell=0}^{L-1} w_\ell \cdot N_\ell \ , \tag{4.10}$$

[4] An original version of this paper more related to the computer vision area is Grauman and Darrell (2005)

where N_ℓ is the number of new approximate matches on level ℓ weighted by level dependent weights w_ℓ. The weights are important, because an approximate match on a finer level should have a stronger influence on the similarity value.

Let $\mathbf{H}(\mathcal{L}) = \left(\boldsymbol{h}^{(0)}(\mathcal{L}), \ldots, \boldsymbol{h}^{(L-1)}(\mathcal{L})\right)$ be the collection of histograms resulted from each quantization $q^{(\ell)}$, *i.e.*

$$h_j^{(\ell)}(\mathcal{L}) = \left|\left\{k \,|\, q^{(\ell)}(\boldsymbol{l}^{(k)}) = j\right\}\right| \quad . \tag{4.11}$$

The minimum intersection kernel is a convenient way to express the number of approximate matches on a certain level:

$$I_\ell \stackrel{\text{def}}{=} \sum_{j=1}^{n_q^{(\ell)}} \min\left(h_j^{(\ell)}(\mathcal{L}^{(i)}), h_j^{(\ell)}(\mathcal{L}^{(i')})\right) \quad , \tag{4.12}$$

where we skipped the dependency on $\mathcal{L}^{(i)}$ and $\mathcal{L}^{(j)}$. We use the formal definition $I_{-1} = 0$ which is useful for expressing the number N_ℓ of new matches at level ℓ as follows:

$$N_\ell = I_\ell - I_{\ell-1} \quad , \tag{4.13}$$

for $0 \leq \ell \leq L-1$. With this definition we have $N_0 = I_0$, which means that every approximate match on the coarsest level is new. We are now ready to combine everything and derive a direct expression for the pyramid match kernel:

$$K^{\text{PMK}}(\mathcal{L}^{(i)}, \mathcal{L}^{(i')}) = \sum_{\ell=0}^{L-1} w_\ell \left(I_\ell - I_{\ell-1}\right) \tag{4.14}$$

$$= w_{L-1} \cdot I_{L-1} + \sum_{\ell=1}^{L-2} \left(w_\ell - w_{\ell+1}\right) I_\ell \quad , \tag{4.15}$$

where we have rewritten the alternating sum to a computationally more efficient expression that directly uses the values I_ℓ of the minimum intersection kernel. The weights are positive and can be moved inside of the minimum expression in Eq. (4.12), which allows writing Eq. (4.15) as a single evaluation of the minimum kernel with suitably weighted feature vectors. An important benefit of the minimum kernel is its ability to exploit sparse features. Entries of BoV histograms are build by only a few elements, which leads to a high degree of

sparsity especially for higher levels in the quantization pyramid. The result of Eq. (4.15) shows that the pyramid match kernel can be represented as a weighted sum of minimum intersection kernels. Therefore, it satisfies the Mercer condition (Theorem 2.3) if we use non-increasing weights w_ℓ, such as $w_\ell = 2^{-\ell}$ as proposed by Grauman and Darrell (2007). With a different number of local features for each image, the PMK formulation of Eq. (4.15) favors larger sets. Therefore, Grauman and Darrell (2005) suggest normalizing the kernel value as follows:

$$\tilde{K}^{\text{PMK}}(\mathcal{L}^{(i)}, \mathcal{L}^{(i')}) = \frac{K^{\text{PMK}}(\mathcal{L}^{(i)}, \mathcal{L}^{(i')})}{\sqrt{K^{\text{PMK}}(\mathcal{L}^{(i)}, \mathcal{L}^{(i)}) \cdot K^{\text{PMK}}(\mathcal{L}^{(i')}, \mathcal{L}^{(i')})}} \ . \tag{4.16}$$

The PMK framework can be justified from a theoretical point of view by bounding the pyramid match value K^{PMK} in terms of the optimal matching cost (4.9) as done by Grauman and Darrell (2007).

4.3.2 Spatial Pyramid Matching Kernel

The PMK framework presented in the previous section allows comparing sets of features in a very efficient manner. For image categorization applications, we can consider comparing sets of local descriptors and use the resulting kernel for classifiers such as SVM (Section 2.5) or GP approaches (Section 2.6.1). In the previous section, we followed Grauman and Darrell (2007) and demonstrated how to build a quantization pyramid for local descriptors neglecting location information completely.

The *spatial pyramid matching kernel (SPMK)* as proposed by Lazebnik et al. (2006a) utilizes the PMK framework to compute a rough geometric correspondence between two images by building a quantization pyramid for local descriptors and their corresponding positions. To be more precise, the quantization of local descriptors is fixed and realized by some clustering method such as the one described in Section 4.2.2. The quantization of the image positions arises naturally and is done by recursively dividing the image into 2×2 or 3×1 cells (van de Sande et al., 2010). The PMK histograms $\boldsymbol{h}^{(\ell)}(\mathcal{L})$ of each level ℓ are simply concatenations of the BoV histograms computed in each cell of the image decomposition. By utilizing these histograms, the calculation of the kernel value is straightforward and equivalent to the pyramid match kernel (4.15). Figure 4.4 shows an example of the image decomposition and the resulting histograms for

the special case of the *pyramid of histograms of orientation gradients (PHoG)* kernel. The visualized principles also hold for SPMK.

He et al. (2008) extend the SPMK framework by learning the weights w_ℓ from available training data. A more general theoretical framework including PMK and SPMK is presented and analyzed by the work of Vedaldi and Soatto (2008), which also proposes a kernel based on matching arbitrary planar graphs rather than predefined grid structures. Bo and Sminchisescu (2009) use kernels to compare single descriptors, which can be seen as a soft matching approach.

Pyramid of Histograms of Orientation Gradients The PHoG kernel of Bosch et al. (2007) can be regarded as a special case of SPMK. Instead of utilizing a codebook estimated from training examples, the PHoG kernel uses histograms of gradient orientations in the same fashion as done by the SIFT descriptor (Section 4.1.2) or the original *histogram of oriented gradients (HOG)* descriptor of Dalal and Triggs (2005). There are some minor implementation differences, such as the use of a binarized gradient image, which can have a high impact on the resulting recognition performance. We refer to Bodesheim (2011) for a further discussion on these issues. For our experiments, we use the original implementation of Bosch et al. (2007).

4.3.3 Real-time Capability

Our bag of visual words framework is able to work close to real-time (around 5 frames per second) for small image sizes (320×240 pixels). To achieve this performance, we use the GPU implementation of SIFT provided by Wu (2007). Clustering SIFT descriptors with a random decision forest only takes a small amount of computation time. Even the the building process of the forest can be done in a few seconds when distributing learning of the trees to different cores. Finally, calculating the minimum intersection kernel highly benefits from using sparse vector representations.

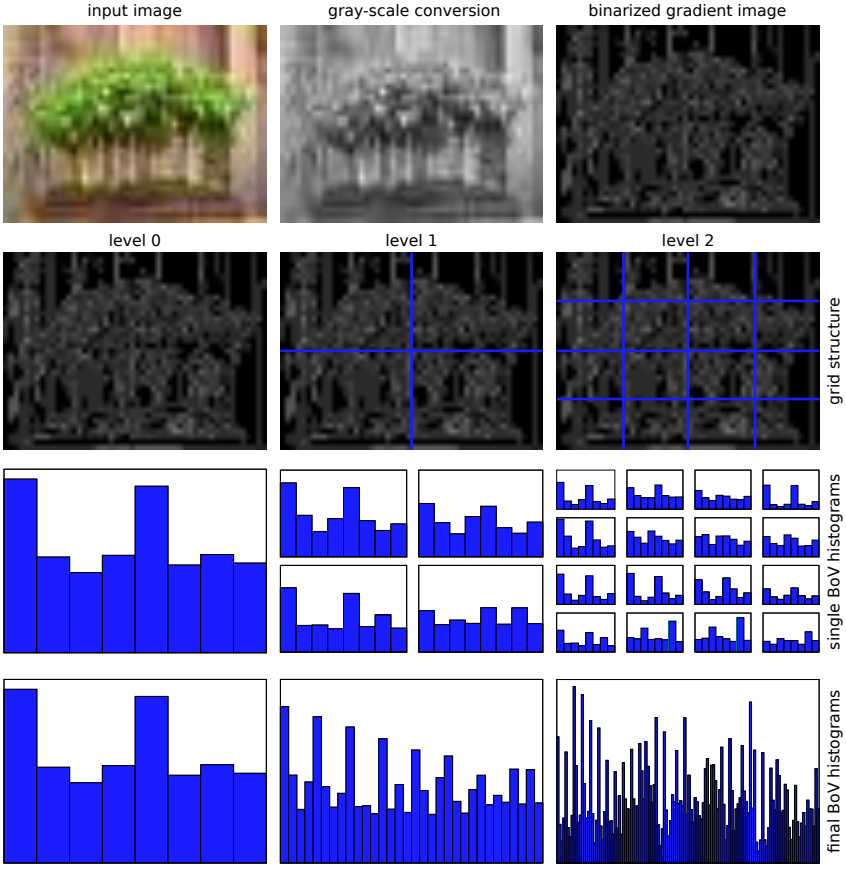


Figure 4.4: Visualization of the pyramid of histograms of orientation gradients (PHoG) approach (Bosch et al., 2007). The rows show results of the main steps involved: (Row 1) a binarized gradient image is calculated; (Row 2) the image is divided into several cells; (Row 3) a gradient orientation histogram is calculated for each cell; (Row 4) all cell histograms of a level are concatenated. Steps (2) to (4) follow the spatial pyramid matching approach presented in Section 4.3.2 *(figure adapted from Bodesheim (2011))*

Chapter 5

Experiments and Applications

In the following chapter, all presented algorithms and approaches are evaluated empirically and compared to previous methods. The evaluation methodology and the used datasets are described in Section 5.1 and are based on state-of-the-art benchmarks. Our transfer learning methods are analyzed in Section 5.2 and Section 5.3 for binary transfer learning with a predefined support task and with automatic support task selection. Section 5.4 studies multi-class transfer learning experimentally. Furthermore, we show the applicability of our one-class classification methods for different visual recognition tasks like image categorization (Section 5.5), action detection (Section 5.6), and defect localization (Section 5.7). In addition, we demonstrate the benefits of the usage of a time-of-flight camera for generic object recognition (Section 5.8).

5.1 Evaluation Methodology

In the following, we briefly present different performance measures used to evaluate our algorithms in subsequent sections. The selected evaluation strategies are commonly applied in current research and have shown to be valuable tools to compare different machine learning systems.

The performance always depends on the selected training and test set and the

variability can be rather high depending on the difficulty of the task. Therefore, we randomly select Z training sets, which are in general overlapping, and use all other examples of the dataset for testing. This procedure allows us to utilize more robust averaged performance values. The corresponding standard deviations can be used for testing the statistical significance.

5.1.1 Multi-class Classification

Let us consider a multi-class classification task with M classes or categories. Every definition of a classification performance value starts from the *confusion matrix* $\mathbf{C} \in \mathbb{N}^{M \times M}$, which is defined as follows:

$$C_{ij} \stackrel{\text{def}}{=} \left| \left\{ \boldsymbol{x} \in \mathcal{D}^{\text{test}} \mid \boldsymbol{x} \text{ belongs to class } j \text{ but was classified as } i \right\} \right| \quad . \tag{5.1}$$

Furthermore, let us use n_j^{t} as a notation for the number of examples of class j present in the test dataset. It can be derived from the confusion matrix by computing column sums, *i.e.* $n_j^{\text{t}} \stackrel{\text{def}}{=} \sum_{i=1}^{M} C_{ij}$. The total number of examples in $\mathcal{D}^{\text{test}}$ is denoted as n^{t}. The most common measure is the *overall recognition rate*, which simply counts the number of correct classification results:

$$\text{err-ov} \stackrel{\text{def}}{=} \frac{1}{n^{\text{t}}} \sum_{j=1}^{M} C_{jj} = \frac{\text{trace}\left(\mathbf{C}\right)}{n^{\text{t}}} \quad . \tag{5.2}$$

A severe disadvantage of this evaluation metric arises in situations with an imbalanced test dataset, *i.e.* when the variability of n_j^{t} is rather high. For example, if we have a test dataset of 990 examples of class 1 and 10 examples of class 2, a trivial classifier, simply assigning every example to class 1, would achieve an overall recognition rate of 99.0%. Due to this reason, we utilize the *average recognition rate*, which averages all class-specific recognition rates (or *hit rates*) in the following manner:

$$\text{err-avg} \stackrel{\text{def}}{=} \sum_{j=1}^{M} \frac{C_{jj}}{n_j^{\text{t}}} \quad . \tag{5.3}$$

This measure is unbiased and every category has the same impact on the recognition rate. Both types of recognition rates can also be used to evaluate binary classification tasks. However, average and overall recognition rates assume equal

misclassification costs and costs related to the empirical class distribution in the training set, respectively (Provost et al., 1998). Both choices might not fit well to the application scenario under consideration. Therefore, we use different performance measures to evaluate algorithms for binary classification tasks.

5.1.2 Binary Classification

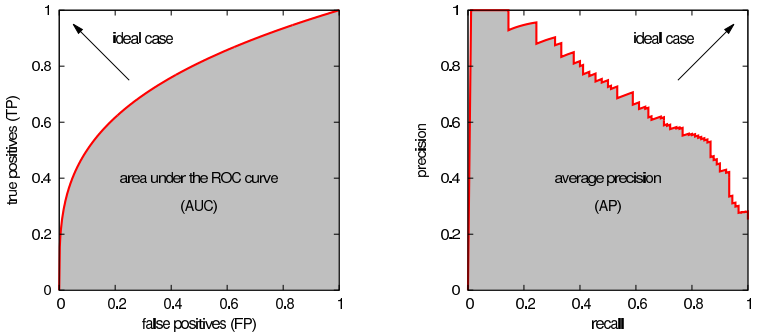
As suggested by Provost et al. (1998), binary classification systems should not be evaluated by accuracy or recognition rates, because those measures can be misleading and are inherently using a fixed cost model for misclassifications. In this thesis, we mainly use *receiver operator characteristic (ROC)* curves, which allow evaluating the performance of an algorithm independent of a cost model. In a binary classification task, we have a 2×2 confusion matrix and we use the following notations:

$$\mathbf{C} = \begin{bmatrix} \text{TN (true negatives)} & \text{FN (false negatives)} \\ \text{FP (false positives)} & \text{TP (true positives)} \end{bmatrix} . \tag{5.4}$$

The prerequisite of utilizing ROC curves is the availability of a classification score rather than a simple discrete decision. Thus, there are multiple ways to do a classification decision based on thresholding the score. For each threshold t, we can compute the *false positive rate* and the *true positive rate* as follows:

$$\text{FPR}(t) = \frac{\text{FP}(t)}{n_{\text{neg}}} , \qquad \text{TPR}(t) = \frac{\text{TP}(t)}{n_{\text{pos}}} , \tag{5.5}$$

with n_{pos} and n_{neg} being the number of positive and negative examples, respectively. Considering all possible thresholds, and the corresponding true positive and false positive rate, yields a two-dimensional curve, which is called ROC curve. An optimal performance would be reached, if the ROC curve passes the point $(0, 1)$ and thereby achieving zero false positives and 100% true positives. However, comparing curves can be difficult and sometimes it is beneficial to boil done everything to a single number. The *area under the ROC curve (AUC)* provides such a measure. An example can be found in Figure 5.1(a). The AUC value also determines the probability of having a positive example $\boldsymbol{x}_1$ and a negative example $\boldsymbol{x}_2$ with the correct relation of the classification score, *i.e.* $f(\boldsymbol{x}_1) > f(\boldsymbol{x}_2)$.



(a) ROC curve (b) Recall-precision curve

Figure 5.1: Performance measures for binary classification: (a) receiver operator characteristic (ROC) curves with the area under the ROC curve (AUC) criterion shown as a shaded area; (b) precision-recall curves with the average precision shown as a shaded area.

An alternative technique to evaluate binary classification tasks is to use *precision-recall curves (PR curves)*, which are defined similar to ROC curves:

$$\text{recall}(t) = \text{TPR}(t) = \frac{\text{TP}(t)}{n_{\text{pos}}} \qquad \text{precision}(t) = \frac{\text{TP}(t)}{\text{TP}(t) + \text{FP}(t)} \quad . \tag{5.6}$$

Note that "recall" is just another synonym for true positive rate. Plotting both measures for different thresholds leads to a curve similar to Figure 5.1(b) with a characteristic saw-tooth shape. The optimal working point is at $(1, 1)$ and the average precision, which is computed as the area under the curve, reduces everything to a single performance measure. The advantage compared to ROC curves is that a precision-recall analysis is more suitable for large-skew class distributions, *i.e.* a setting with a large variability in prior class probabilities (Davis and Goadrich, 2006). Although there are some interesting and tight connections between the AUC measure and average precision, they do not necessarily lead to the same performance ranking of algorithms (Davis and Goadrich, 2006).

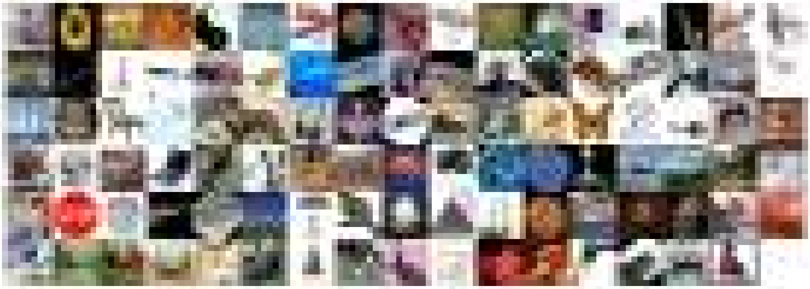
5.1.3 Datasets for Visual Object Recognition

In the main part of our experiments, we use different types of publicly available image databases and categorization benchmarks, which are briefly described in the following. Example images of each dataset are given in Figure 5.2. We use different experimental data for wire rope analysis, action detection, and generic object recognition with multiple sensors. Therefore, details as well as some example images are included in the corresponding sections.

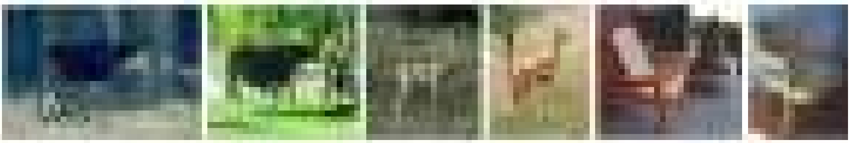
Handwritten Latin Letters The database of Fink (2004) is a collection of images containing handwritten Latin letters, resulting in 26 object categories. For each category, 60 binary images are provided. Every image has a size of 35×35 pixels. The main advantage of this database is its simplicity, which allows using very basic feature extraction methods. Optical character recognition applications are in general suitable for studying transfer learning algorithms, because letters often share important similarities and common image transformations.

Caltech-101 and Caltech-256 The Caltech-101 database was introduced by Fei-Fei et al. (2006) and is one of the most famous benchmarks for categorization approaches. It consists of 100 object categories and one background category. Images have been downloaded with image search engines and partly processed afterward to have one dominant orientation of objects within a category (Fei-Fei et al., 2006). Griffin et al. (2007) extended the database resulting in Caltech-256. There are several advantages of both databases. The most important one is the variability of the object categories. As can be seen in Figure 5.2(a), the database consists of photos, drawings, and symbols. This property is ideal to test transfer learning algorithms that automatically select support tasks, such as our GP approach presented in Section 3.4. Another advantage is the presence of a background category, which allows for easily testing of one-class and binary classification methods. Furthermore, the Caltech-256 has been used for evaluating transfer learning methods in the work of Tommasi et al. (2010) and Tommasi and Caputo (2009). Therefore, performing experiments with this database allows for direct comparison. However, this database suffers from a severe dataset bias, which is elaborated and discussed in Section 5.1.4.

Birds and Butterfly Dataset In addition to the already presented datasets, we combined the bird and butterfly datasets used in Lazebnik et al. (2004) and



(a) The Caltech-101 database [illegible] represented by one image of each category



(b) [illegible]

(c) Images of the Caltech-256 database [illegible]

(d) [illegible]

(e) [illegible]

Figure 5.2: Example images of datasets used for experimental evaluation in the current chapter.

Lazebnik et al. (2006b). The goal is to create a categorization benchmark with object categories that can be divided into two different semantic sets. It should be noted that we do not compete with the results of Lazebnik et al. (2006b), because their work focuses on recognition with a part and constellation approach rather than on image categorization with global BoV features.

5.1.4 Discussion about other Databases and Dataset Bias

The works of Ponce et al. (2006) and Torralba and Efros (2011) discuss the bias of certain image databases. As argued by the authors of both papers, the Caltech-101 database is not suitable to train recognition systems that are aimed to work in complex real-world applications and scene understanding tasks. One important reason is that object images of this database were mainly captured from online shops, which show the object in a center position and often in front of artificial backgrounds. This leads to a bias, which has been already observed in Figure 3.8. Furthermore, due to the post-processing applied by Fei-Fei et al. (2006), images of some categories (*e.g. airplanes* and *motorbikes*) show a characteristic image border, which can be exploited by classification systems to distinguish them easily from other categories.

We use the Caltech databases without these critical categories to evaluate our transfer learning algorithms due to the reasons given in Section 5.1.3. Additionally, other object databases are often not directly suitable to assess transfer learning algorithms. For example, the Pascal VOC databases (Everingham et al., 2010) only comprise twenty object categories, which is not sufficient to evaluate the performance in heterogeneous transfer learning situations. Furthermore, the difficulty of this database shifts the focus towards the feature extraction rather than the underlying machine learning algorithms. The mammals database of Fink and Ullman (2008) was directly designed to serve as a benchmark for visual transfer learning algorithms. However, it severely suffers from label noise and has been used only by a small number of researchers.

A suitable alternative database for evaluating transfer learning algorithms would be the ImageNet database of Deng et al. (2009), which was recently released to compare large-scale classification approaches. Nonetheless, up to now our GP approaches do not scale to millions of training images. Further information about this research topic is given in Section 6.2.

5.2 Binary Transfer Learning with a Predefined Support Task

The following section studies binary transfer learning and aspects of the proposed methods. As defined in Section 3.1, the aim of binary transfer learning is to use prior knowledge from previously learned binary classification tasks to support the learning of a new target task. In this section, we concentrate on transfer learning with a single predefined support task. Automatic selection of support tasks with dependent Gaussian processes will be evaluated in Section 5.3. The experimental setup is briefly described in Section 5.2.1. Furthermore, we evaluate our relevance transfer method in detail in Section 5.2.2. A comparison of all three methods presented in this thesis for binary transfer learning is given in Section 5.2.3.

5.2.1 Experimental Dataset and Setup

We use image data from the Caltech-101 dataset (Fei-Fei et al., 2006) to show the applicability to image categorization tasks. Four classes are used to conduct binary classification tasks: *okapi*, *gerenuk* and *chair* vs. the Caltech background class. The training set consists of a varying number of images from the object category and 200 training images of the background class. Figure 5.2(b) shows example images of each category. Performance is measured with the area under the ROC curve averaged using the results achieved from 10 random selections of the training set.

Feature extraction is done by computing a bag of visual words (BoV) representation of OpponentSIFT features as described in Chapter 4. We use binarized BoV histograms and a codebook of 1500 visual words generated with the method described in Section 4.2.2.

The dependent Gaussian process method, which is evaluated in Section 5.2.3, utilizes a Gaussian kernel with hyperparameter $\gamma = \exp(-2.5)$. For the regularized tree method, we set the hyperparameter σ^2 of the constrained Gaussian prior to 0.001, which is not optimized with respect to the test set. Parameter values of the randomized decision forest classifier, are chosen according to the standard values given in Table B.2.

5.2.2 Evaluation of Feature Relevance Transfer

In preliminary experiments, we evaluate our transfer learning approach presented in Section 3.3, which transfers information about the relevance of features. The results of the experiments can be summarized as follows:

1. Transferring feature relevance from related tasks helps to improve the recognition performance in the case of few training examples.

2. The benefit is most prevalent, if the support task is visually similar to the target task.

3. A maximum a posteriori estimation of the probability of feature relevance is not necessary and we can use a flat prior with $\beta = 1$.

Dirichlet Parameter In a first experiment, we evaluate the influence of the prior distribution in Eq. (3.23) (Section 3.3.5). This data-independent prior distribution serves as a smoothing term for the estimation of relevant features. We build a random decision forest using a fixed set of 30 examples of the *okapi* class and 200 examples of the background class. By using the learned forest, we estimate feature relevance as described in Section 3.3.5 with a varying Dirichlet parameter β. The estimates are used to perform transfer learning with our feature relevance method utilizing a fixed training example of the *gerenuk* class and a set of background images. The classifier is evaluated on all remaining images of the *gerenuk* task. Average performance values and the corresponding standard deviation of $Z = 50$ runs are illustrated in Figure 5.3(a).

It can be seen that with an increasing value of β the performance drops and the optimal value remains at $\beta = 1$. For this reason, we fix β to this value, which corresponds to maximum likelihood estimation of $\boldsymbol{\theta}$. This highlights that the complete removal of features that are irrelevant for the support task with $p(g_i \in \mathcal{R}) = 0$, is beneficial. This may not be the case in situations with a smaller feature set and features that are completely irrelevant for the support task but essential for a target task.

Influence of the Ensemble Size We analyze the influence of the number T of base classifiers in the ensemble. The same experiment as in the previous paragraph is performed with a varying size of the ensemble. The results are illustrated in Figure 5.3(b).

The performance increases with the number of base classifiers in the ensemble, which was also motivated theoretically in Section 2.2.3. Another interesting effect is that the performance benefit of transferring feature relevance is most prevalent when a small number of base classifiers are used. For further experiments, we utilize random decision forests with 200 trees.

Different Support Tasks and Levels of Transfer In the following, we study the performance of the binary classification task "*gerenuk* vs. *background*" with different types of knowledge transfer:

1. transfer of feature relevance using the support classes *okapi* and *chair*,

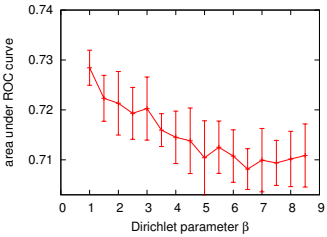
2. using the BoV codebook of the support class without transferring feature relevance information, and

3. no knowledge transfer, *i.e.* random decision forests (Section 2.3) applied to the few examples of the target task.

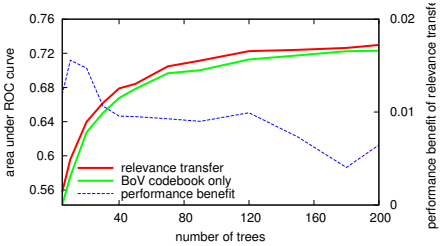
Note that the latter variant also means that a codebook is generated only from training examples of the target task. Figure 5.4(a) illustrates the resulting recognition rates. Additionally, Figure 5.4(b) shows the same results for the class *okapi* with prior information learned from the *gerenuk* task.

At a first glance, it can be seen that transferring feature relevance from related tasks improves the recognition performance compared to learning without knowledge transfer. This performance benefit is most prevalent with a visually similar class, such as the corresponding related animal class. Using prior knowledge from the chair *class* is in some cases also beneficial. It is most likely that this is due to learning of a natural generic prior knowledge, which also showed in other work to improve the recognition performance (Fei-Fei et al., 2006). An additional discussion about generic prior knowledge is included in Section 5.4.4.

Transferring only the BoV codebook of the support task also increases the performance. The difference between our feature relevance approach and this method in Figure 5.4(a) for one training example might seem to be minor and insignificant, due to a standard deviation of about 1% in the previous experiment (Figure 5.3(a)). However, using a paired t-test and the corresponding average results of all 10 training and test runs, we are able to show significance with a level of $p < 0.003$.

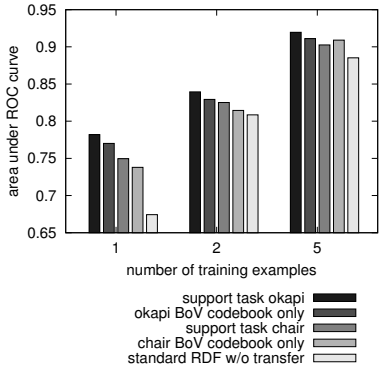


(a) Evaluation of the Dirichlet hyperparameter β

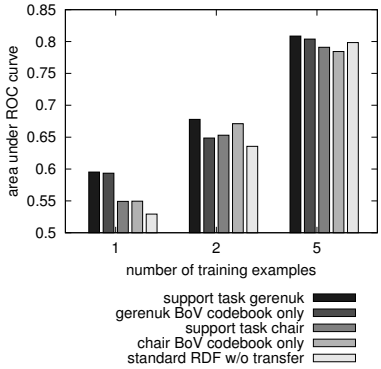


(b) Evaluation of the ensemble size

Figure 5.3: Influence of the (a) hyperparameter β of the Dirichlet distribution, which is used to smooth the probabilities of feature relevance, and the (b) number of decision trees in the forest on the transfer learning performance.

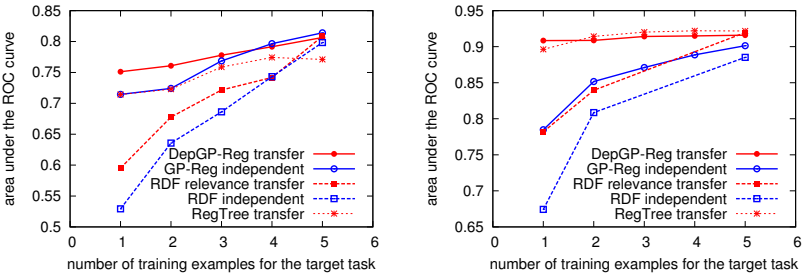


(a) Target task *gerenuk*



(b) Target task *okapi*

Figure 5.4: Experiments with the target task (a) *gerenuk* and (b) *okapi* vs. *background* using different support tasks and a varying number of training examples of the target task.



(a) Target task *okapi* (b) Target task *gerenuk*

Figure 5.5: Comparison of all three transfer learning approaches presented in this thesis: Transfer learning with regularized trees (RegTree, Section 3.2), feature relevance (Section 3.3) and dependent Gaussian processes (DepGP-Reg, Section 3.4). The graphs show the performance of the classifiers on the target task *okapi* and *gerenuk* utilizing the support tasks *gerenuk* and *okapi*, respectively.

5.2.3 Comparison of all Methods for Binary Transfer

In the following experiments, we compare all presented transfer learning methods applied to transferring knowledge between binary classification tasks. As a target task, we utilize the object categories *okapi* and a varying number of training examples. We also use the predefined support task *gerenuk* and the corresponding BoV codebook obtained from 30 training examples. Another experiment is derived by exchanging target and support task. The results are depicted in the two plots of Figure 5.5. An evaluation of the automatic selection of support tasks with our dependent GP method will be done in Section 5.3.

First of all, let us have a look on the performances of independent learning. Blue graphs (the plot is best viewed in color) show the performance of standard RDF and a GP classifier using label regression, which do not transfer any knowledge from the support task. It can be seen that the GP method outperforms the RDF classifier, however, both methods achieve a comparable performance for 5 training examples.

Let us first have a look on the results of the first experiment in Figure 5.5(a). Our dependent GP approach achieves the best performance. Note that in spite of a predefined support task, the method still determines the task correlation

Table 5.1: Comparison of the transfer learning methods developed in this thesis.

	Advantages	Disadvantages
Dependent Gaussian processes	+ allows for automatically selecting the support task and the degree of transfer + achieves the best recognition performance + suitable for all kernel functions	- cubic runtime $\mathcal{O}(n^3)$ for learning - runtime for classifying one test example depends on the number of support training examples
Feature relevance transfer	+ can be easily integrated in current systems + no additional computational load compared to independent RDF learning	- worst performance in our binary transfer experiments
Regularized decision trees	+ pre-build decision trees are reused, which allows for online learning on a category level + allows for multi-class transfer	- hyperparameter selection of the prior is crucial - overfitting effect for multi-class transfer (Section 5.4)

parameter automatically. The regularized tree method results in a lower performance but compared to the feature relevance method it performs significantly better. Furthermore, every transfer learning method improves the area under the ROC curve compared to the corresponding baseline approach (RDF or GP). In addition, the regularized tree method does not converge directly to independent learning with the RDF classifier. The results of the second experiment in Figure 5.5(b) are similar. The most interesting difference is that the dependent GP approach and the regularized tree method achieve a comparable performance for $n > 1$.

Table 5.1 compares all three methods with respect to their advantages and disadvantages.

5.3 Heterogeneous Transfer Learning

The following experiments study the suitability and different aspects of our transfer learning method based on dependent Gaussian processes (Section 3.4). In contrast to the previous section, we consider sets of binary classification tasks with very different characteristics and try to select a suitable support task automatically. We investigate the influence of our WordNet pre-selection method (Section 3.4.4) and compare our approach to two state-of-the-art transfer learning techniques. The results of the experiments can be summarized as follows:

1. Our transfer learning approach improves the performance on average compared to independent learning, even with a large heterogeneous set of available support classification tasks (Section 5.3.3).

2. By using WordNet pre-selection, we can achieve a performance gain for nearly all classification tasks (Section 5.3.3).

3. Our method achieves higher recognition rates than Adapted LS-SVM (Tommasi and Caputo, 2009) and its extended version (Tommasi et al., 2010) in the case of unrelated object categories (Section 5.3.2).

4. Transfer learning with GP classification (Laplace approximation) improves with respect to independent GP classification but fails to be competitive with the label regression approach even for independent learning (Section 5.3.3).

5. Using the average precision of leave-one-out estimates as described in Section 3.4.3 yields the best performance among several other model selection criteria (Section 5.3.3).

5.3.1 Experimental Datasets and Setup

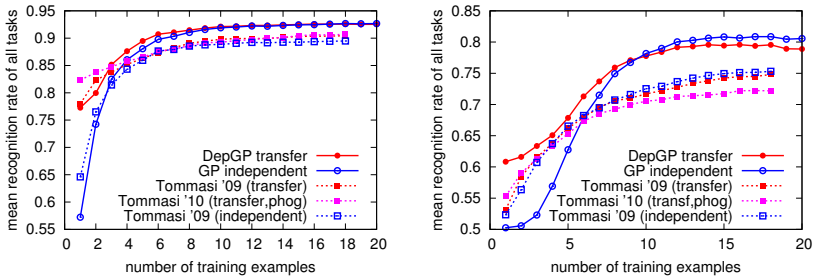
Experiments are performed using all 101 object categories of Caltech-101 and a subset of the Caltech-256 database (Section 5.1.3). Both databases contain a large number of challenging object categories and a suitable background category. In each experiment a target task and few training images are selected. Training and testing is done for each target task 100 times with a random split of the data, which yields mean performance values. The performance of our transfer learning method is compared to independent learning, which uses GP regression with training images of the target task only.

5.3.2 Experiments with Caltech-256

We compare our approach using label regression (DepGP-Reg) to Adapted LS-SVM as proposed by Tommasi and Caputo (2009) and Tommasi et al. (2010). Therefore, we use the experimental setting described in Tommasi and Caputo (2009) (Caltech-256 subsets, PHoG features) and the software provided by Tommasi et al. (2010), which includes both versions of Adapted LS-SVM. Two sets of classification tasks are chosen to study the cases of (1) transferring knowledge using only related support classification tasks (*car*, *fire-truck* and *motorbike*) and (2) using a heterogeneous set (*school-bus*, *dog* and *duck*). Training and testing is done with a variable number of training images for the target object category and 18 training images for the background and support categories. As a performance measure, we use the mean recognition rate of all tasks averaged over all 100 random selections of the training data.

Note that in our publication in Rodner and Denzler (2010), we compare our results to the ones provided in Tommasi and Caputo (2009, Figure 1(a) and 2(a)), which differ from the figures obtained with the provided software and used in the following analysis. This might be due to the manual selection of images done by Tommasi and Caputo (2009), which lead to a subset of images where the object is clearly visible without occlusion.

We also compare our approach to Tommasi et al. (2010), which allows transferring from several support tasks in combination. The authors also use



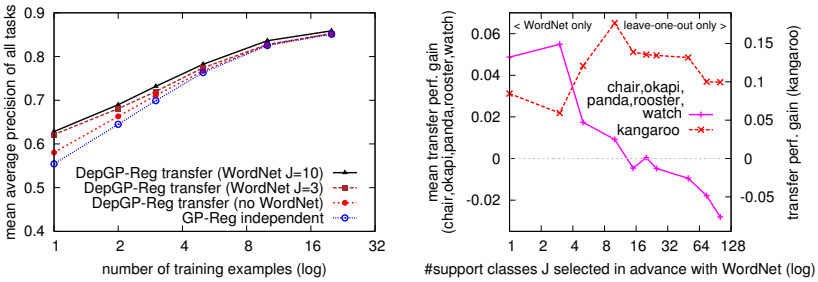
(a) *Related* classification tasks (b) *Unrelated* classification tasks

Figure 5.6: Caltech-256 results of our transfer learning approach, independent learning and Adapted LS-SVM of Tommasi and Caputo (2009) and its extension proposed in Tommasi et al. (2010).

multiple kernels to boost the recognition performance. However, we restrict our experiments to a single kernel to focus explicitly on aspects of transfer learning. Therefore, we limit the approach of Tommasi et al. (2010) to PHoG features (Section 4.3.2) using the minimum intersection kernel (Section 2.4.2).

A pre-selection of classification tasks using WordNet is not applied in this experiment.

Evaluation The results are shown in Figure 5.6. First of all, it is clearly visible that learning benefits from knowledge transfer. Figure 5.6(b) also validate that we are able to improve the results of Tommasi and Caputo (2009) and Tommasi et al. (2010) in the "unrelated case", in spite of achieving lower recognition rates for our baseline approach without knowledge transfer. The interesting observation is that in contrast to the previous approaches, our method is able to extract suitable prior knowledge from the given support tasks. For learning with few training examples and with given "related" support tasks, we achieve a comparable performance to Tommasi and Caputo (2009) but inferior results compared to Tommasi et al. (2010), which transfers information from all support tasks at once. Furthermore, the classification performance of the GP framework converges to higher values than the SVM approach.



(a) Average performance (b) Detailed WordNet analysis

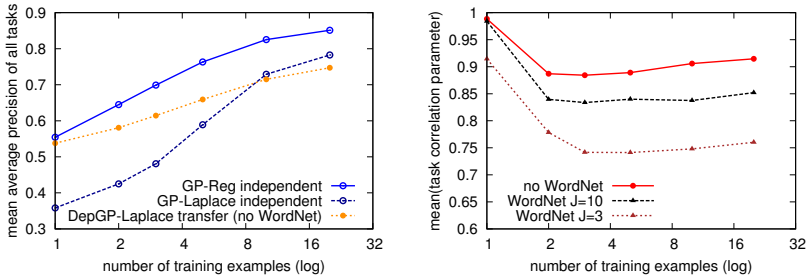
Figure 5.7: (a) Mean average precision of all tasks with a varying number of training examples. (b) Different peaks of the one-shot learning performance for a varying number of support classes J pre-selected using WordNet: Mean average precision of tasks, which did not benefit from knowledge transfer without WordNet, and performance values of the *kangaroo* task for different values of J. The results of the *kangaroo* task highlights the importance of the combination of WordNet and our model-selection.

5.3.3 Experiments with Caltech-101

In these experiments, we use all 101 object categories as available support tasks and a subset of possible target tasks (listed in Figure B.3(a)). As a performance measure for each binary classification task, we use average precision as used in the Pascal VOC challenges (van de Sande et al., 2010). Training and testing is done with a variable number of training images of the target object category, 30 training images of the support categories and 200 background images. The kernel function is computed using the SPM framework applied to OpponentSIFT features and a visual codebook of 1500 elements (Chapter 4).

5.3.3.1 Evaluation and Comparison with Independent Learning

As can be seen in Figure 5.7(a), our transfer learning approach without WordNet pre-selection improves the mean average precision compared to independent learning when using few training examples and converges to it with more than 10 training examples. Furthermore, Figure 5.8(a) shows the results of GP classification with Laplace approximation for independent (GP-Laplace) and transfer learning (DepGP-Laplace). The interesting observation is that independent GP



(a) GP classification vs. regression (b) Analysis of the estimated ρ

Figure 5.8: (a) Comparison of GP regression and classification (probit model, Laplace approximation) for independent and transfer learning. (b) Analysis of the average value of the task correlation parameter ρ with respect to the number of training examples (DepGP-Reg).

classification with Laplace approximation achieves a significant worse performance in the case of few training examples compared to the label regression approach. Transfer learning with dependent GP improves the performance in a large extent, but still is unable to reach the performance of independent GP regression. This result is in accordance with the statement of Bishop (2006, p. 216) that Laplace approximation needs a lot of data to provide good solutions. Due to this reason, we restrict the following experiments to label regression.

The detailed results for each task using a single training example are included in the appendix in Figure B.3(a) and deliver additional insight into the methods behavior. Transfer learning improves the average precision for some tasks significantly, *e.g.* task *gerenuk* with a performance gain of more than 11%, but also fails for some other tasks like *okapi*. This is due to a wrong selection of the support task using leave-one-out estimates and can be handled in almost all cases by using the WordNet pre-selection method. Our transfer learning method fails for the task *watch*, because there seems to be no related task in the database. Figure 5.7(b) shows the benefit of WordNet for those cases by varying the number J of pre-selected support tasks. The same plot also highlights that severe pre-filtering with WordNet ($J < 10$) leads to worse results for the task *kangaroo*. The same holds for the mean average precision of all tasks which is lower for a strict pre-selection (with $J = 3$) compared to a pre-selection of only

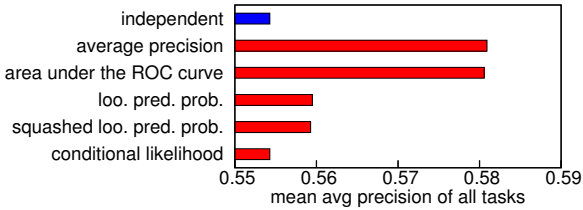


Figure 5.9: Mean average precision of all tasks using a single training example without pre-selection and different model selection criteria (DepGP-Reg).

10 support tasks (*cf.* Figure 5.7(a)). Therefore, only a combination of WordNet pre-selection with a selection based on leave-one-out estimates is reasonable when confronted with a new task. Further investigation of the relationship between both selection criteria is done in Section 5.3.3.3.

We additionally evaluated our approach with different model selection criteria: average precision and area under the ROC curve calculated with leave-one-out estimates, leave-one-out predictive probability (Rasmussen and Williams, 2005) with squashed variants (Tommasi and Caputo, 2009), and the conditional likelihood of the target task training set (Cao et al., 2010). The results are shown in Figure 5.9, justifying our choice of average precision using leave-one-out estimates.

5.3.3.2 Analysis of the Adaptation of the Task Correlation Parameter

In Section 3.4.1, we showed how the task correlation parameter controls the assumed correlation between the latent functions associated with a task and the degree of transfer. Whereas for few training examples, we expect the optimal task correlation parameter ρ to be around $\rho = 0$, it should decrease when the size of the training set increases.

This intuition is verified in Figure 5.8(b), which plots the average of the estimated value of ρ with respect to the training set size. The average is taken over all tasks and random splits of the dataset corresponding to the previous experiment. The mean value of the task correlation parameter indeed decreases as expected in the first part of the graph. However, there is also a small increase when using more than five training examples. This effect might be due to overfitting to non-appropriate support tasks and will be further studied in

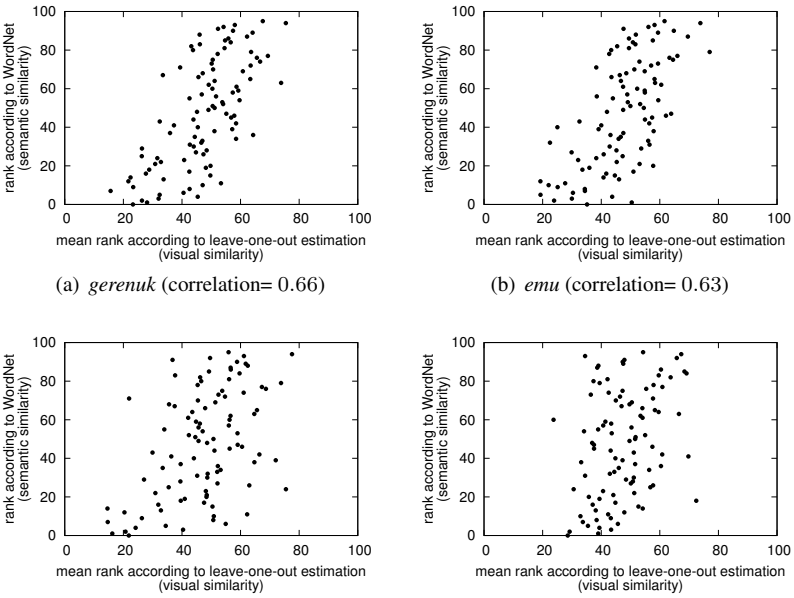
Section 5.4.

In the case of a single training example, the average task correlation parameter is $\rho \approx 1$, which means that all training examples of the support task are directly used as training examples of the target task (*cf.* Section 3.4.1). Another interesting observation can be done by comparing the graphs of different values of the number of categories J pre-selected by WordNet. Transfer learning without WordNet pre-selection ($J = 100$) utilizes higher values of ρ than our method with $J = 10$ or $J = 3$. The pre-selection efficiently reduces the risk of overfitting and yields more conservative and proper values of the task correlation parameter yielding to a better recognition performance in general as we have seen in previous plots.

5.3.3.3 Correlation between Semantic and Visual Similarity

In the following experiment, we analyze the correlation of the results of the two selection criteria used in our approach: the leave-one-out criterion based on information present in the given training examples, and semantic similarities computed using WordNet. The question is how visual similarity and semantic similarity correlate.

In Figure 5.10, we compare the ranking of support tasks estimated by leave-one-out and WordNet by computing the mean rank position. In this experiment, we use four different target tasks with one training example. The degree of correlation, computed with the Spearman test, highly depends on the target task under consideration. It varies from 0.34 for the *chair* task, which can be seen as uncorrelated, to 0.66 for the *gerenuk* task, which is often interpreted as moderate correlation. In theory, WordNet pre-selection should be more beneficial if the results of both selection criteria are not correlated. By considering the performances achieved by each of the four tasks (Figure B.3(a)), we can see that this statement holds. The paper of Deselaers and Ferrari (2011) contains a similar analysis of the relationship between semantic and visual similarity. Their study is based on the ImageNet database and averages over all suitable pairs of categories.



(a) *gerenuk* (correlation= 0.66)

(b) *emu* (correlation= 0.63)

(c) *okapi* (correlation= 0.45)

(d) *chair* (correlation= 0.34)

Figure 5.10: Analysis of the ranking of support tasks derived from leave-one-out estimation and semantic similarities computed using WordNet. Four target tasks are considered with one training example and the correlation coefficient calculated by the Spearman test is given in the captions.

5.4 Multi-class Transfer Learning

In the following section, we analyze our regularized decision tree approach for multi-class transfer learning (Section 3.2.5). In contrast to binary transfer learning, we consider a multi-class classification task that contains a target class with few training examples. The goal is to improve the recognition task of the complete task by exploiting similarities of the target class and a set of support classes.

In our experiments, we support the following hypotheses:

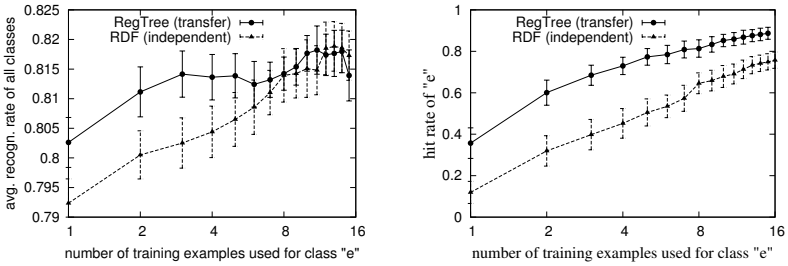
1. Regularized trees lead to a significant performance gain for multi-class classification with few training examples (Section 5.4.2).

2. Our method uses prior knowledge that relies on visual similarity, and is thus not related to generic prior knowledge (Section 5.4.3).

5.4.1 Experimental Dataset and Setup

The evaluation criteria are the unbiased average recognition rate of the whole classification task (Section 5.1.1) and single recognition rates of the target class. Our experiments aim to analyze the gain of our transfer learning approach compared to the random decision forest classifier (Section 2.3). For this reason, our choice of features is not optimized. The variance σ^2 of the prior and the RDF parameters are set to the values as in the experiments described in Section 5.2.1. Furthermore, we select support classes manually. In a real-world application, support classes could be selected using the WordNet method presented in Section 3.4.4.

Letter Recognition We test our approach for the task of handwritten letter recognition and use the database of Fink (2004), which was already presented in Section 5.1.3. Classification is done with an ensemble of 10 decision trees and the following scenario is selected: target class *e* and support classes *a,b,c,d*. The images in this database are binary, therefore, a very simple feature extraction method is used. The whole image is divided into an equally spaced $w_x \times w_y$ grid. In each cell of the grid, the ratio of black pixels to all pixels within the cell is used as a single feature. This leads to a feature vector with $w_x \cdot w_y$ dimensions. In all experiments, the values $w_x = 8$ and $w_y = 12$ are used.



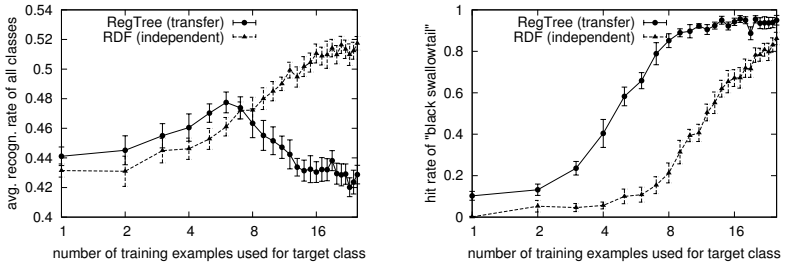
(a) Average recognition rate (b) Hit rate

Figure 5.11: Comparison to independent learning of RDF classifiers in a multi-class classification task using the letter recognition dataset of Fink (2004). The left plot shows the average recognition rate of the whole classification task with respect to the number of training examples (log scale) of a specific class. On the right side the single recognition rate of this class is plotted.

Image Categorization To demonstrate the behavior of the method on a high-level image categorization task, we combine the birds (Lazebnik et al., 2006b) and the butterflies dataset (Lazebnik et al., 2004) into one single multi-class classification task (Section 5.1.3). Thus, the object categories can be divided into two different semantic sets. The category *black swallowtail* is used as a target class τ, and all other butterfly categories serve as support classes $\mathcal{S}$. Training data consists of a variable number of training images for τ and 26 images for each of the remaining classes. This classification task is more difficult than our letter recognition setting. For this reason, an ensemble of 500 decision trees is used. We also use rgSIFT features and a large BoV codebook of 13000 elements computed as described in Section 4.2.2.

5.4.2 Evaluation of Multi-class Performance

The results of the main experiment evaluating the overall multi-class classification performance can be found in Figure 5.11 and 5.12. The plots show the average recognition rate of the whole task (plots on the left side) and the recognition rate of the target class (plots on the right side) compared to those of the original random decision forest method.



(a) Average recognition rate (b) Hit rate

Figure 5.12: Comparison to independent learning of RDF classifiers within a high-level multi-class classification task using the bird-and-butterfly dataset as used in Lazebnik et al. (2004, 2006b). Semantic of the plot is analogous to Figure 5.11.

It can be seen that our method improves the recognition rate of the target class and the average recognition rate in the range with few training examples (1 to 8 examples). The regularization is therefore able to transfer knowledge from support classes without violating the ability of distinguishing between classes.

After a specific number of training examples, the average recognition rate decreases while the recognition rate or hit-rate of the target class (plots on the right side) still grows. The effect corresponds to over-regularization. The influence of the prior distribution is controlled by the hyperparameter σ^2, which is kept to a fixed value independent of the training examples used. Therefore, the MAP estimation of leaf probabilities leads to many leaves with non-zero posterior probabilities of the target class. This corresponds to a large variance of the distribution in feature space, which dominates the distribution of all other classes. The variance of the class distribution reaches a critical threshold leading to an overestimation of the distribution corresponding to the target class. The classifier prefers the target class, which results in a worse average recognition rate (or an increasing number of false positives) of the whole classification task.

It should be noted that this phenomenon is typical for the application of transfer learning methods in a multi-class classification task. In contrast, binary transfer learning algorithms converge to the performance of independent learning after a specific number of training examples, which is due to the treatment of a support and target class as independent binary classification tasks. A similar

effect is observed in the context of zero-shot learning (Rohrbach et al., 2010b, Figure 3).

5.4.3 Similarity Assumption

What happens if support classes are selected that do not share common features with the target class? As mentioned in our introduction, the concept of transfer learning is based on the main assumption that support classes $\mathcal{S}$ are somehow similar to the target class τ. Therefore, it is possible to further assume that those similarities can be captured in feature space by a distribution $p(\boldsymbol{\theta})$ of the classifier parameter $\boldsymbol{\theta}$. The following experiment tries to uncover whether the knowledge transferred is related to a generic or a more category-specific prior. A category-specific prior concentrates on transferring more detailed elements, such as object parts. To answer this question, an experiment using the letter recognition scenario is performed. As a target class with a weak representation of 4 training examples, we select the letter *e* and use two different sets of similar support classes (*a*,*b*,*c*,*d*) and dissimilar support classes (*m*,*n*,*w*,*v*,*z*).

Figure 5.13 shows a scatter plot of several runs, where each point corresponds to the average recognition rate of a random decision forest classifier without (ML estimation) and with our transfer learning method (MAP estimation). All points above the diagonal indicate a clear benefit from prior knowledge. It can be seen that visually dissimilar classes (triangular dots in red color) do not lead to a performance gain and can even decrease the performance (negative transfer).

5.4.4 Discussion of the Similarity Assumption

The results clearly show that our transfer learning method learns prior knowledge that is not related to generic prior knowledge. This is an important difference to a lot of other approaches that capture generic prior knowledge. For example, in Fei-Fei et al. (2006), MAP estimation is applied to transfer knowledge between object categories such as: *motorbikes*, *faces*, *airplanes* and *wild cats*. Therefore, their method seems to use a generic prior of object category images (*e.g.* exploiting that the size and location of objects are not uniformly distributed). Bart and Ullman (2005) also test their approach with a large set of various unrelated categories of the Caltech-101 database and showed that the knowledge transferred, which is represented by shared image fragments, helped to improve the recognition performance. In general, the use of generic prior knowledge

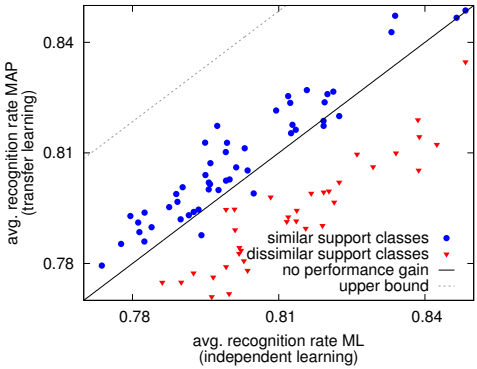


Figure 5.13: Average recognition rate of the ML approach in comparison to the rate after applying our MAP re-estimation technique. The regularization results in a performance gain only if support classes are (visually) similar to the target class.

has its own tradition and motivation, especially in the context of natural image statistics (Torralba and Oliva, 2003). In our opinion the use of category-specific in addition to generic priors is essential to capture available knowledge as much as possible, and thus allows efficient learning with few examples, similar to the development of the human visual system.

5.5 Object Recognition with One-Class Classification

In the following section, we empirically analyze our GP-based one-class classification (OCC) methods presented in Section 3.5. As already stated, this is joint work with *Michael Kemmler*, who did an important part of the following experiments. The application we are considering is the task of binary image categorization or object detection as already studied in previous sections about transfer learning. The main difference is that we do not use examples of a background (negative) category for learning, and we directly apply one-class classification methods to the images of single object categories. This setting allows us to analyze the behavior and performance of OCC methods for several

binary classification tasks. The experimental setup is described in Section 5.5.1 and we compare our GP-based novelty scores with the SVDD method (Section 3.5.5.1) and study the influence of kernel hyperparameter changes. In the following analysis, we empirically support the following hypotheses:

1. The negative predictive variance estimated by GP regression (GP-Reg-V), used as a novelty score for OCC, significantly outperforms all other methods using a color image kernel (Section 5.5.1) and it achieves a better performance than SVDD for various values of the outlier ratio ν (Section 5.5.2).

2. GP regression outperforms approximate GP classification with Laplace approximation (LA) and expectation propagation (EP) for OCC tasks (Section 5.5.2).

3. The performance of the predictive mean of GP regression (GP-Reg-M) as a novelty score varies dramatically for different object categories and can even decrease with an increasing number of training examples (Section 5.5.3).

4. Introducing additional kernel hyperparameters with parameterized image kernels allows boosting the performance with the disadvantage of additional experimental parameter tuning (Section 5.5.4).

5. Nonlinear OCC techniques with image kernels are able to learn the appearance of specific object attributes (Section 5.5.5).

5.5.1 Experimental Dataset and Setup

Experiments are performed with all object categories of the Caltech-101 database created by Fei-Fei et al. (2006) and described in Section 5.1.3. All algorithms are evaluated using the area under the ROC curve (AUC), which is estimated by averaging the results of 50 random splits in training and testing data. Whereas the training sets only consist of 15 images of a single object category, testing data contains the remaining images of the category and all images of the Caltech background category. We utilize two image kernels: the PHoG kernel of Bosch et al. (2007) (Section 4.3.2) and a spatial pyramid matching kernel build from OpponentSIFT features (Section 4.1.3 and 4.3.2). We refer to the latter one as color image kernel.

Table 5.2: Mean AUC performance of all OCC methods, averaged over all 100 binary classification tasks. Bold font is used when all remaining measures are significantly outperformed. GP measures significantly superior to SVDD$_\nu$ (with optimal ν) are denoted in italic font.

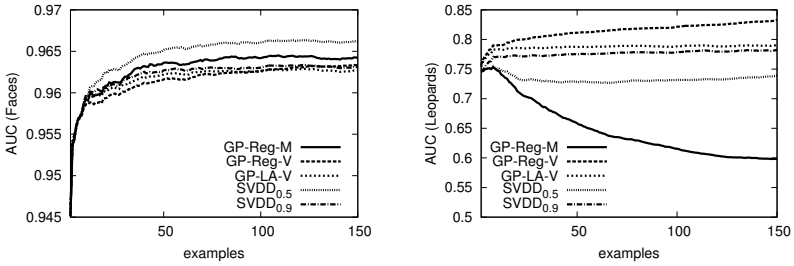
	GP-Reg-P	GP-Reg-M	GP-Reg-V	GP-Reg-H	GP-LA-P	GP-EP-P
PHoG	*0.696*	0.693	0.692	*0.696*	0.684	0.683
Color	*0.761*	0.736	**0.766**	*0.755*	*0.748*	*0.747*
	GP-LA-M	GP-EP-M	GP-LA-V	GP-EP-V	SVDD$_{0.5}$	SVDD$_{0.9}$
PHoG	0.684	0.683	0.686	0.685	0.690	0.685
Color	0.745	0.744	*0.758*	*0.757*	0.739	0.746

5.5.2 Evaluation of One-Class Classification Methods

Let us first have look on the overall performance of all methods. We compare all combinations of novelty scores and inference algorithms: predictive probability (-P), mean (-M) and variance (-V) estimated by GP regression (GP-Reg) and GP classification using Laplace approximation (GP-LA) or expectation propagation (GP-EP). Furthermore, we analyze the heuristic $\mu_* \cdot \sigma_*^{-1}$ for GP regression (GP-Reg-H) and compare all of our methods with SVDD using several outlier fractions $\nu \in \{0.1, 0.2, \ldots, 0.9\}$ (SVDD$_\nu$). An overview of our OCC novelty scores can be found in Table 3.1. We use 15 randomly chosen examples for training and average the AUC value of all binary classification tasks derived from the Caltech-101 database and all 50 random selections of the training images. Table 5.2 shows the results for both image kernels (PHoG and color). We only give the values of the best performing SVDD measures.

It can be seen that PHoG features are significantly inferior to color features for this task. Therefore, we restrict our experiments in subsequent sections to the color-based image kernel. The average performances of all measures are similar, but SVDD is significantly outperformed by at least two GP measures for all tested values of the outlier ratio ν (t-test, $p \leq 0.025$). The best performance values are achieved by the predictive variance of GP regression (GP-Reg-V), with significantly higher AUC values than all other methods using color features. For the PHoG kernel, GP-Reg-V achieves a comparable performance to SVDD for any tested value of ν.

Another interesting result is that GP classification with a probit error model



(a) *faces* category (b) *leopards* category

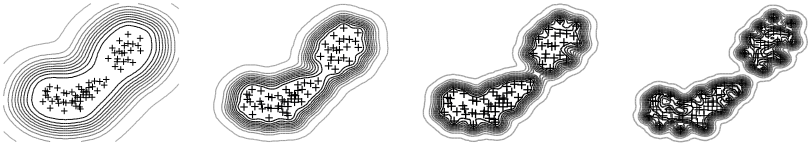
Figure 5.14: Results for OCC object recognition of the categories *faces* and *leopards* with a varying number of training examples and the color kernel.

(Section 2.6.6) does not improve the OCC performance, even though the model seems to be more adequate from a theoretical point of view. Laplace approximation and expectation propagation are outperformed by the measures GP-Reg-V and GP-Reg-P derived from GP regression.

5.5.3 Performance with an Increasing Number of Training Examples

We now analyze the behavior of all methods for an increasing number of training examples. The plots in Figure 5.14 show the performance of OCC methods applied to the categories *faces* and *leopards* with respect to the training set size. Whereas Figure 5.14(a) exhibits a normal behavior of an increasing generalization ability, the performance of the *leopards* classifier decreases for some methods when providing more training examples. Especially the predictive mean method (GP-Reg-M) suffers from this problem, but also SVDD shows a similar inability to learn with increasing training sets.

What seems to be unusual for a learning algorithm, is in general a severe underlying problem of OCC methods and density estimation algorithms. As pointed out in Section 3.5.5.2, consistency of one-class SVM and GP-Reg-M equipped with a Gaussian kernel is guaranteed only if the hyperparameter γ is decreasing for an increasing amount of training data. The parameter γ controls the smoothness of the predicted distribution and we refer to hyperparameters



(a) $\gamma = 50$ (b) $\gamma = 200$ (c) $\gamma = 500$ (d) $\gamma = 1000$

Figure 5.15: Influence of the hyperparameter γ of the Gaussian kernel function on the shape of the OCC score function (GP predictive variance). The training set is displayed with crosses and level sets of the score function are shown.

with a similar property as smoothness parameter. In the current experiment, we used a fixed image kernel function and the underlying smoothness parameter was not changed. Therefore, no consistency of the learning algorithm is guaranteed and the performance indeed drops for an increasing amount of training data. Note that this undesirable property is not restricted to our methods and applies to OCC algorithms in general.

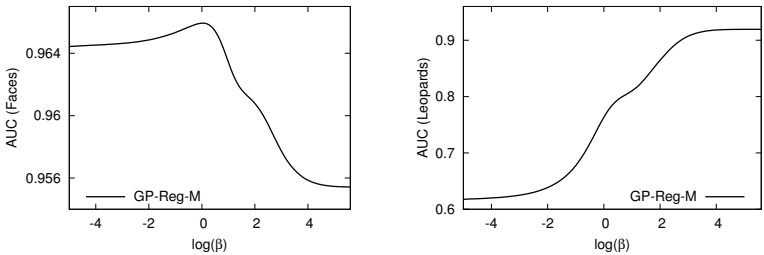
5.5.4 Influence of an Additional Smoothness Parameter

As pointed out in the previous section, estimating the correct degree of smoothness of the predicted distribution is one of the key problems in OCC and density estimation. The smoothness is often controlled by tuning the hyperparameter of a parameterized kernel, such as the Gaussian kernel. An example can be seen in Figure 5.15. We use image kernels, which are not parameterized at all. The decreasing performance of the predictive mean method in the last experiment might be due to this inflexibility.

In the following, we want to further investigate this behavior. Therefore, we parameterize our image kernel function by transforming it into a metric, which is then plugged into a generalized radial basis function kernel (Definition 2.4 and Vedaldi and Soatto (2008)):

$$K_\beta(\boldsymbol{x}, \boldsymbol{x}') = \exp\left(-\beta\left(K(\boldsymbol{x}, \boldsymbol{x}) - 2K(\boldsymbol{x}, \boldsymbol{x}') + K(\boldsymbol{x}', \boldsymbol{x}')\right)\right) \quad . \tag{5.7}$$

Experiments are performed with the modified kernel function K_β utilizing 100 training examples and a varying value of β. The results for the categories *faces* and *leopards* are given in Figure 5.16.



(a) *faces* category (b) *leopards* category

Figure 5.16: Influence of an additional smoothness parameter β of a modified image kernel function on the OCC performance for the object categories *faces* and *leopards*.

Let us first have a look on Figure 5.16(b) and the results for the category *leopards*. The performance for a small value of β is comparable to the non-parameterized version (*cf.* Figure 5.14(b)). However, increasing the parameter value leads to a performance above 0.9, which is superior to all other methods, such as the predictive variance method (GP-Reg-V). The same plot for the category *faces* highlights that the optimal selection of this parameter highly depends on the task because a completely different range of parameter values leads to high performance values. Right after the displayed points, we ran into severe numerical problems in both settings due to small kernel values below double precision. We expect a similar behavior of approximate GP classification methods when tuning the scale parameter of the cumulative Gaussian noise model (Section 2.6.6). Our analysis shows that introducing an additional smoothness hyperparameter offers a great potential, though optimization beyond simple cross-validation with some negative training examples is still an unsolved problem.

5.5.5 Qualitative Evaluation and Predicting Category Attributes

In the following experiment, we apply our OCC methods to the difficult task of estimating a membership score of a specific sub-category or category attribute. We train our GP methods and SVDD with 30 images of a type of chair called

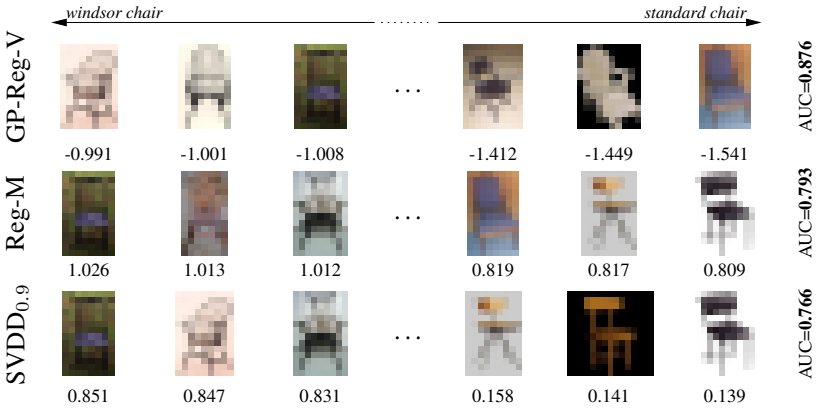


Figure 5.17: Results obtained by training different OCC methods for *windsor chair* and separating against *chair*: the three best ranked images (all of them are characteristic windsor chairs) and the three last ranked images with corresponding output values are shown.

windsor chair, which has a characteristic wooden backrest. The performance is tested on all remaining *windsor chairs* and images of the category *chairs*. The upper and lower ends of the estimated ranking are illustrated in Figure 5.17. The qualitative results are similar for all methods, but the AUC values clearly show that GP-Reg-V is superior.

5.6 Action Detection

In the following section, we apply our one-class classification methods to *action detection* tasks. The aim is to detect actions, such as *walking*, *hand-waving*, and *running*, in a given video sequence. As already stated in Section 1.5, one of the advantages of OCC methods is that there is no need to directly model the background class with training examples, *i.e.* with video sequences without the corresponding action. This property is beneficial, if the variety of the background class is rather large. Although our experiments are restricted to action detection, the applied machine learning methods could also be used for *event detection*, *i.e.* detecting novel complex actions in a video stream (Zhao et al., 2011; Oh et al., 2011).

The outcomes of the experiments can be summarized as follows:

- Our image categorization framework presented in Chapter 4 can be easily extended to action detection and categorization in video streams.
- The performance of action detection with one-class classification highly depends on the task and the selected hyperparameter.

5.6.1 Experimental Dataset and Setup

For our experiments, we utilized the KTH database (Schuldt et al., 2004). The dataset consists of videos of six action categories performed by different actors. As can be seen in the example images given in Figure B.1, the background is static and homogeneous. Therefore, compared to other action datasets, such as the Hollywood dataset of Laptev et al. (2008), the recognition scenario is simple. However, our goal is to study the applicability of OCC methods rather than focusing on action recognition, which is a research area on its own and planned to be a future research topic. We use the predefined split in training and test sets as given in the description of the database. This results in 64 short videos of an action category for learning.

In Chapter 4, we presented how to perform image categorization with the bag of visual words (BoV) framework. The BoV principle has also shown to be suitable for action recognition tasks (Laptev, 2005; Zhang et al., 2008; Niebles et al., 2008). Due to this reason, we directly apply the same ideas to extract features in videos for action recognition.

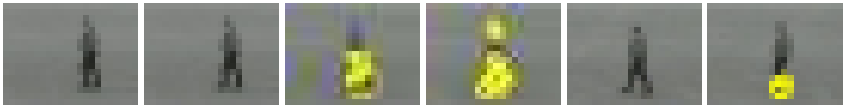


Figure 5.18: Examples of space-time interest points detected in a video sequence belonging to the category *walking* of the KTH database (Laptev, 2005).

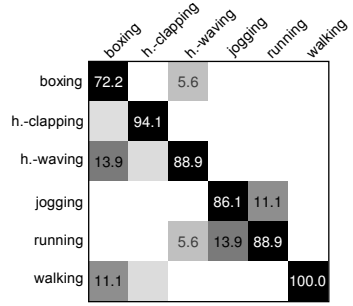
As local features, we use the *space-time interest points (STIP)* and local spatio-temporal descriptors presented by Laptev (2005) [1]. Both approaches can be directly seen as extensions of the Harris corner detector and the histogram of oriented gradients descriptor (Section 4.3.2) to the time domain. We skip the technical details related to those methods and provide a visual impression in Figure 5.18. All obtained local features are used to build a visual codebook with the method of Moosmann et al. (2006b) (Section 4.2.2). Furthermore, kernel values are computed utilizing the SPMK approach (Section 4.3.2) restricted to quantizations of the image domain, which has the effect that we do not incorporate any information about the temporal order.

5.6.2 Multi-class Evaluation of our Action Recognition Framework

To assess our action recognition framework, consisting of the choice of local features and kernel functions, we apply it to the categorization task usually studied with the KTH database. The objective is to distinguish between several action categories. The results are shown in Figure 5.19.

The right table in Figure 5.19 additionally gives the results of several other authors who use the same database. In spite of using a simple BoV framework and without further tuning of the used features, our approach achieves an average and comparable performance. The confusion table on the left side of Figure 5.19 shows some typical failures of the system, such as confusing *jogging* with *running*. Furthermore, *boxing* seems to be a mixture of *hand-waving* and *walking*, which is quite intuitive.

[1] In our experiments, we make use of the software provided by Ivan Laptev at `http://www.irisa.fr/vista/Equipe/People/Laptev/download.html`.



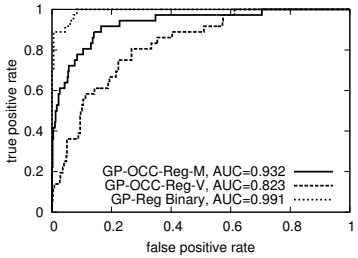
Reference	Recognition Rate
Schuldt et al. (2004)	71.1%
Niebles et al. (2008)	83.3%
Wong and Cipolla (2007)	86.7%
Savarese et al. (2008)	86.8%
our simple BoV approach	88.4%
Kläser et al. (2008)	91.4%
Laptev et al. (2008)	91.8%
Han et al. (2009)	94.1%

Figure 5.19: Comparison of our simple action recognition framework to previous work on action categorization with the KTH database. The left image shows the achieved confusion matrix and only values below 3% are displayed.

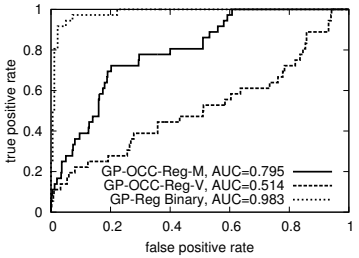
5.6.3 Evaluation of One-class Classification Methods

We now turn to the problem of action detection. Six different OCC tasks from the KTH database are derived by selecting an action category as a positive class and using all video sequences of the other categories as negative examples. The selection of the training examples directly follows the standard experimental setup described in Schuldt et al. (2004), which leads to 64 short training videos for each category. We evaluate one-class classification with GP regression utilizing the predictive mean and variance (Section 3.5.2), as well as binary classification with GP regression with additionally using negative examples for learning. The results are given in Figure 5.20, which shows for each action detection task the corresponding ROC curves for all three methods.

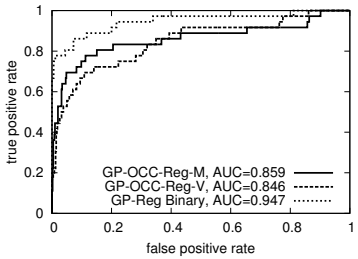
At a first glance, it can be seen that one-class classification can not improve the performance obtained by binary classification with GP regression. This is a common effect for datasets with a low variability of the negative class, which can be learned easily from some training examples. Another important observation is that the gap between the performance of one-class classification compared to the binary case significantly varies across all tasks. On the one hand, we have a similar performance of all three methods for the task *running* with an AUC value around 0.92. On the other hand, the performance of our OCC method utilizing the variance criterion of Gaussian process regression is not better than random chance for the task *hand-clapping*. This is due to the



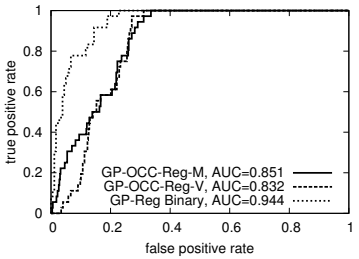
(a) *boxing*



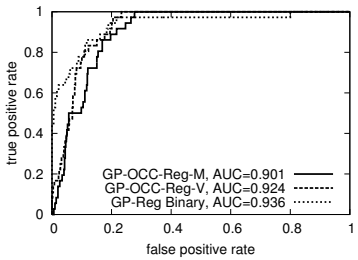
(b) *hand-clapping*



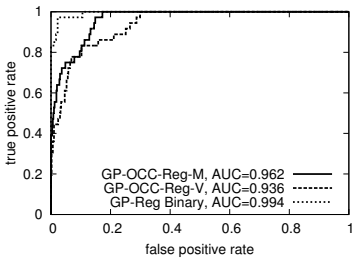
(c) *hand-waving*



(d) *jogging*



(e) *running*



(f) *walking*

Figure 5.20: Results of our OCC methods and GP regression applied to the binary tasks derived from the six action categories of the KTH database. All plots show ROC curves for one-class classification with GP (mean and variance criterion) and binary classification with GP regression.

common hyperparameter selection problem already studied in Section 5.5. The results of OCC methods highly depend on hyperparameter selection and the values of optimal hyperparameters depend on the task.

As a summary, we can say that one-class classification can be suitable for action detection but care has to be taken with hyperparameter selection. We expect that the advantages compared to binary classification are more prominent when being confronted with more complex scenes, which will be a future research task.

5.7 Defect Localization with One-Class Classification

In this section, we apply our one-class classification methods (Section 3.5.2) to a defect localization task. The work has been published in Rodner et al. (2011) and was done together with *Esther Wacker*, who provided pre-computed features and her expertise in automatic wire rope analysis.

Automatic visual inspection is one of the key industrial application areas of computer vision and machine learning. Especially, the task of defect localization is of high importance due to the increasing demand of fast and reliable quality assurance (Kumar, 2008). We concentrate on automatic visual inspection of wire ropes, which are the basic elements of several important constructions in everyday life: elevators, bridges and ropeways. The reliability of wire ropes is often the most important safety factor. An example of a (simple) surface defect on a wire rope is displayed in Figure 5.21. Standard approaches for automatic quality control are magnetic measurement techniques (Zhang et al., 2006). However, these methods are unable to detect defects on the wire rope surface and due to this reason, additional visual inspection is necessary. Manual visual analysis of wire ropes is a difficult, exhausting, and error-prone procedure, which has to be done by experts trained to recognize different error types, like broken wires and lightening strokes. In contrast, automatic defect detection allows processing wire ropes with minimal human intervention and constant performance. Capturing rope images is done by a system of four line cameras similar to the one presented in Moll (2003) and depicted in Figure 5.22.

There are two main challenges for defect detection systems operating on wire ropes. The first issue is a severe lack of training examples for defects. Images of real-world defects are difficult to obtain and collecting suitable training examples

is often impossible. Another problem arises from the variability of defects and their often indistinguishable appearance compared to normal non-defective rope images. If we apply an ordinary supervised learning approach to classify parts of the rope as defective or non-defective, the system would be inherently biased towards the specific appearance of the few training images given as defect examples. A solution to this problem is to utilize one-class classification (OCC) as motivated in Section 1.5 and presented in Section 3.5.2. Our defect localization algorithm is described in Section 5.7.2 and further information about previous approaches for wire rope analysis is given in Section 5.7.1.

The main contributions of this part of the thesis are as follows:

1. We show the suitability of our GP-based OCC methods as presented in Section 3.5.2 for defect localization tasks.

2. The presented approach does not exploit the periodic or specific structure of wire ropes and is therefore applicable to other defect localization tasks.

5.7.1 Related Work on Wire Rope Analysis

We briefly review previous work on wire rope analysis. A literature review of one-class classification techniques can be found in Section 1.6 and a recent and more detailed review of wire rope defect localization is included in Platzer and Denzler (2011).

A computer vision system can easily detect scratches on work pieces (Chin and Harlow, 1982), defects in textured materials (Kumar and Pang, 2002) or abnormalities on food products (Blasco et al., 2007). However, there are only a few published papers on visual analysis of wire ropes. This might be due to the lack of appropriate data and the difficulty of defect localization, which are even challenging for human experts. Our approach is an extension of the work of Platzer et al. (2008, 2009a, 2010), which perform defect localization

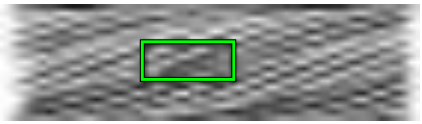


Figure 5.21: A typical surface defect on a wire rope: a broken wire

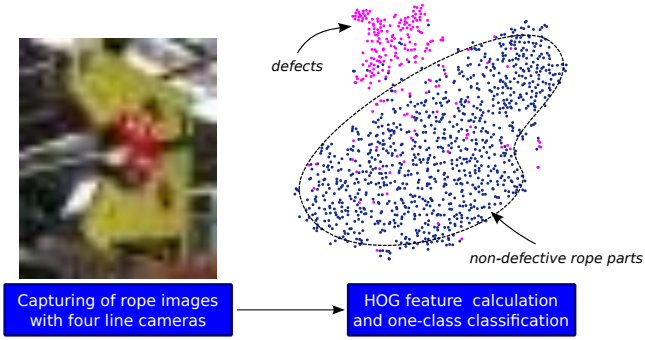


Figure 5.22: Rope images are captured by four line cameras arranged to analyze a wire rope (Moll, 2003). The scatter plot on the right hand side visualizes the high-dimensional histogram of oriented gradients features with the unsupervised t-SNE method of van der Maaten and Hinton (2008). Feature vectors are extracted from real rope data (`ROPE1`, view 3).

by calculating features of the rope images and classifying them with Gaussian mixture models. Their feature sets include HOG features (Section 4.3.2), features derived from linear prediction, and several others. Furthermore, Platzer et al. (2009b) proposes an approach based on hidden Markov models to exploit the periodic structure of the rope. The work of Haase et al. (2010) focuses on combining the information provided in all four views and using the strong correlation structure present in the data.

The advantage of wire ropes is their periodic structure, which can be described with a 3d curve model. Wacker and Denzler (2010) uses such a model to perform analysis-by-synthesis and register the model to the images of real-world wire ropes. Their flexible registration technique allows estimating characteristic rope parameters like lay lengths. A significant change of these parameters could be caused by an internal structural defect, which is impossible to directly observe visually or to detect with magnetic flux methods (Zhang et al., 2006). The registered 3d model of the rope and a learned appearance model also allow precisely detecting visual surface defects (Platzer and Denzler, 2011).

The aim of the following study is to show how well one-class classification can work in this scenario, without using any context knowledge or structural information of wire ropes. This allows us to develop a generic method, which is

suitable for a wide range of other defect localization applications. We compare our results to those obtained by the generative Gaussian mixture model described by Platzer et al. (2010), which also does not exploit context information.

5.7.2 Anomaly Detection in Wire Ropes

Our defect localization system follows a simple image processing pipeline. We subsequently calculate features of the rope image and apply a one-class classification approach, like our method proposed in Section 3.5.2. The resulting novelty scores are thresholded to detect defective areas of the rope. Learning consists of training the OCC classifier with a large set of non-defective rope data.

Platzer et al. (2010) evaluates several feature types according to their suitability for wire rope defect detection. We follow their suggestion and use histograms of oriented gradients (HOG) as introduced by Dalal and Triggs (2005) and reviewed in Section 4.3.2. Broken wires result in strong image edges perpendicular to the usual twist direction and HOG features allow capturing them as characteristic histogram peaks. HOG features are computed using a block of 20 camera lines (image columns) divided into 20×20 cells. Gradient histograms with 4 bins are computed for each of the cells (Section 4.1.2) and combined into one normalized feature vector. Furthermore, we follow Platzer et al. (2010) and add the entropy of the histograms as an additional feature. The entropy allows incorporating information about the presence of dominant orientations in the current image patch. The feature vector has a dimension of $D = 5\frac{h}{20}$ (entropy and 4 orientations for each cell), where h is the rope diameter in pixels. The segmentation of the rope is done as a preprocessing step and h is automatically selected to be the maximum pixel diameter observed at a full lay length of the rope.

5.7.3 Experimental Dataset and Setup

The goal of our experiments is to compare our one-class classification approach (Section 3.5.2) with several other techniques. Therefore, we use two different rope datasets, which are also used in Platzer et al. (2010) and Platzer and Denzler (2011). Acquisition is done using the system of Moll (2003) under completely realistic conditions. In the following, we refer to the two different datasets as `ROPE1` and `ROPE2`. `ROPE1` has a length of approximately 1.3km and `ROPE2` is 400m long. The resolution of the line cameras is known to be 0.1 mm/camera

line. Each dataset was labeled by a human expert.

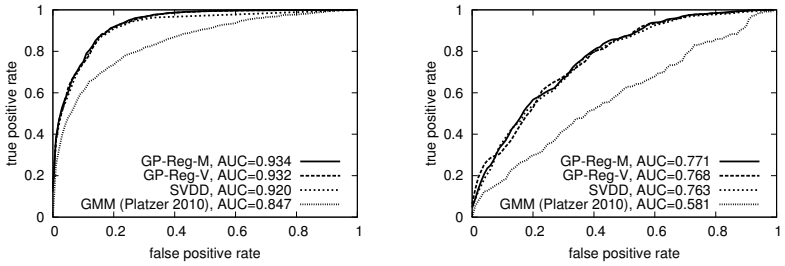
The rope datasets are different in nature with respect to the complexity of the surface defects. ROPE1 contains easy recognizable defects as can be seen in the feature visualization of Figure 5.22. Defects contained in ROPE2 are often inconspicuous, small, and difficult to detect, even for a human expert. We also applied the t-SNE method (van der Maaten and Hinton, 2008) to visualize the features extracted from ROPE2 in a 2d plot. However, there was no clear seperation visible between both classes, which highlights the difficulty of this dataset.

All methods are trained on a rope sequence of 100,000 camera lines (10m rope, 5000 training examples), which was proposed by a human expert as a defect-free rope region. Evaluation is performed on the remaining rope sequence, which contains all labeled defects.

We do not utilize approximate inference techniques for GP classification, because they did not lead to a performance benefit for OCC tasks as observed in the experiments in Section 5.5.4. For our experiments, we use the Gaussian kernel with a standard hyperparameter value of $\gamma = \exp(-2.5)$. An evaluation of the influence of this parameter is given in Section 5.7.4.4. The noise parameter σ_n^2 is automatically determined as described in the end of Section 2.6.10. An important parameter of the SVDD method (Section 3.5.5.1) is the outlier fraction ν, which is also experimentally analyzed, but without a significant difference in the results. Therefore, ν is set to 0.1.

5.7.4 Evaluation

Defect localization is a binary classification task and for our evaluation we utilize ROC curves (Section 5.1) and the corresponding area measure. In a real-world system, a decision threshold has to be experimentally tuned and adapted to the required false positive (false alarm) rate. Positive examples are examples of surface defects and negatives correspond to normal rope data. Note that this terminology is different to the one used in our presentation of one-class classification algorithms and in our experiments in Section 5.5, where we trained an OCC classifier on positive examples rather than on negatives. This change is due to our goal of a comparable framework to previous work on wire rope analysis (Platzer et al., 2010).



(a) ROPE1 (b) ROPE2

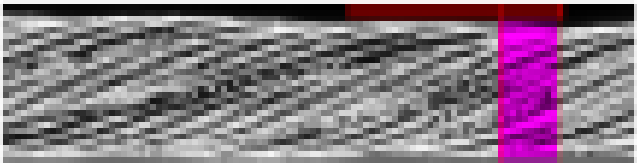
Figure 5.23: Average ROC curves both rope datasets. The curves for all three kernel-based methods (GP mean, -variance and SVDD) are very similar and best viewed in color.

5.7.4.1 Comparison with Other Methods

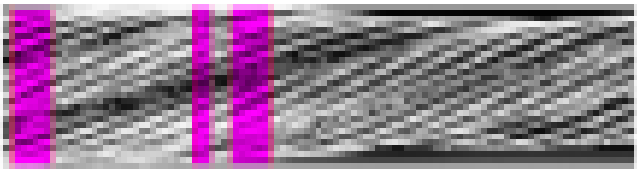
The results obtained for all methods are displayed in Figure 5.23 using ROC curves with the area under the ROC curve (AUC) given in the legend. Note that the ROC curves are averaged over the results obtained for the four individual camera views.

It is obvious that all three kernel-based OCC approaches, predictive mean and variance of GP regression and SVDD, outperform the classical GMM strategy proposed in Platzer et al. (2010). Additionally, the AUC values suggest that the GP-based OCC approaches offer a slightly better performance than SVDD. The predictive mean approach achieves the best results and is also faster than the posterior variance approach during testing (Section 3.5.3). Please note that approaches which exploit the special structure of wire ropes achieve higher recognition results. For example, Platzer et al. (2009b) achieve an AUC value of 0.816 and Platzer and Denzler (2011) is even able to obtain a performance of 0.987 for ROPE2[2]. We especially concentrate on defect localization without any prior knowledge to ensure an universal applicability with respect to other application areas.

[2]Performance values are obtained from Platzer and Denzler (2011).



(a) Correct defect detection of a broken wire.



(b) False positives occurred on a non-defective part of the rope.

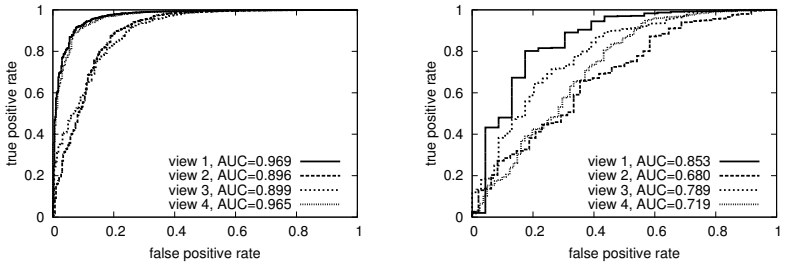
Figure 5.24: Example detections of our approach. Results are highlighted and the bar on top of the rope shows the defect annotation of a human expert.

5.7.4.2 Qualitative Analysis

Figure 5.24 shows some example detections of our algorithm, where wire parts with a prediction mean below a manually selected threshold are recognized as defects. The broken wire in Figure 5.24(a) is correctly recognized (bar in magenta color) and we can see that the size of the ground truth annotation (bar on top of the rope in red color) is unreasonably large. An important open problem is the high number of false positives as can be seen in Figure 5.24(b).

5.7.4.3 Analyzing Individual Views

Let us analyze the performance on all four individual views. Figure 5.25 displays the ROC curves and AUC values for each view and both datasets. Especially for ROPE3, the performance depends on the camera view. Some defects are more prominent in view 1 and 3 and are difficult to recognize in other views. The interesting fact is that for ROPE3 and view 3, we reach a comparable performance (AUC=0.759) to the HMM approach of Platzer et al. (2008b) which achieved an AUC value of 0.798.



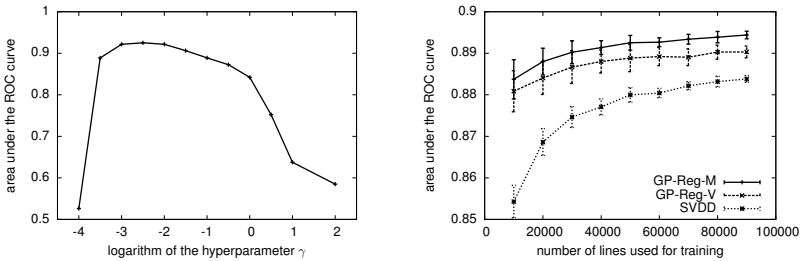
(a) ROPE1 (b) ROPE2

Figure 5.25: ROC curves achieved by GP mean for each view of both rope datasets. The figure is best viewed in color.

5.7.4.4 Parameter Evaluation and Influence of the Training Set Size

In previous experiments, we used a fixed hyperparameter γ. In Section 5.5, we have seen that this parameter can have a severe influence on the performance. Figure 5.26(a) shows the defect detection performance with respect to the value of $\log(\gamma)$ for the first view of ROPE1 and analyzed with the predictive mean method. Although there is a large performance drop below -3.5 and above 0, there is still a wide area with performance values beyond 0.85.

Figure 5.26(b) plots the influence of the number of lines used for training on the performance of the GP predictive mean method working on the second view of ROPE1. The number of training examples n can be derived by dividing this number with 20, because we use a block with a width of 20 pixels to calculate the HOG features. The standard deviation is calculated by using multiple random subsets for training. The interesting fact is the remarkable performance difference of SVDD and our GP-based methods for few training examples.



(a) Influence of the kernel hyperparameter (b) Influence of the training set size

Figure 5.26: Detection performance with a varying (a) value of the hyperparameter and (b) number of training examples. The performance is analyzed on the first and second view of `ROPE1` with GP predictive mean, respectively.

5.8 Learning with Few Examples by Using Multiple Sensors

Research in machine learning, computer vision, and in our previous studies mainly concentrated on image categorization using a single visual sensor or photos from the web (Zhang et al., 2007). Despite the great success of all methods, the importance of depth information for reliable generic object recognition is mostly ignored. In the following work, we present an approach to generic object recognition using the combined information of a visual sensor and range information obtained from a time-of-flight (ToF) camera (Lange, 2000). We show that especially learning with few examples benefits from additional sensor information. The work was done in collaboration with *Doaa Hegazy* and has been published in Rodner et al. (2010).

A ToF camera offers real-time depth images obtained by modulating an outgoing beam with a carrier signal and measuring the phase shift of that carrier when received back at the ToF sensor. Incorporating the advantages of this new camera technology into computer vision systems is current research and is successfully done in 3d reconstruction tasks (Cui et al., 2010) or marker-less human body motion capture (Ganapathi et al., 2010).

We use a ToF camera together with a CCD camera to solve generic object recognition problems. To combine the information of our two sensors, we utilize Gaussian process classification (Section 2.6.1), which allows efficient hyperparameter estimation and integration of multiple sensor information using kernel combination (Section 2.6.10). In contrast to previous work (Kapoor et al., 2010), which use GP regression to approximate the underlying discrete classification problem, we also study Laplace approximation (Section 2.6.7), which directly tackles the discrete nature of the categorization problem. Hyperparameter estimation is done by extending multitask techniques for GP regression to LA as proposed in Section 2.6.10. Additionally, we apply the framework of spatial pyramid matching kernels (Section 4.3.2) to different kinds of local range features. Figure 5.27 presents an overview of the proposed approach and the main steps involved. In summary, the main contributions of this part of the thesis are as follows:

1. We present multiple kernel learning with Gaussian process classification for sensor data fusion (previous research is limited to visual sensors only (Kapoor et al., 2010)).

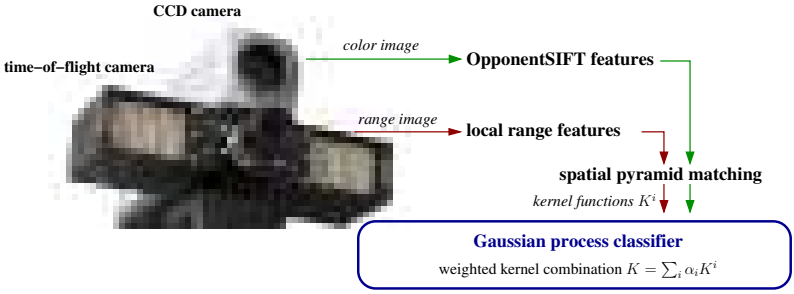


Figure 5.27: An overview of the proposed approach: color and range images are captured using a CCD camera and a time-of-flight camera (PMD Vision Technologies 19k); Local features are extracted from the data of both sensors and "kernelized" using spatial pyramid matching, which yields different kernel functions.

2. Different noise models of GP classification are studied for image categorization and sensor data fusion (previous research is limited to Gaussian noise and GP regression (Kapoor et al., 2010)).

3. Our method yields a significant improvement in terms of recognition performance compared to the current state-of-the-art of object recognition with time-of-flight range images (Hegazy, 2009).

5.8.1 Related Work

In our work, we utilize local range features as presented by the work of Hetzel et al. (2001) and reviewed in Section 4.1.4. They introduce different feature types and similarity measures for histograms and apply nearest neighbor classification. Toldo et al. (2009) suggest applying a preliminary segmentation of complete 3d models and a description of the resulting parts with the bag of visual words idea (Section 4.2). Classification is done with multiple equally weighted histogram intersection kernels and support vector machines. The suitability of SIFT features for range image matching is studied by Zhang and Wang (2009), who propose to compute SIFT features on normal texture and shape index images (Hetzel et al., 2001). In contrast, Lo and Siebert (2009) present an extension of SIFT features suitable for range images. Special feature types for time-of-flight cameras are

studied by Haker et al. (2008). A key idea of their work is the non-equidistant Fourier transformation to represent range images.

Our method for feature calculation is mainly based on spatial pyramid matching (Section 4.3.2) as presented by Lazebnik et al. (2006a) for standard image categorization and used by Li and Guskov (2007) for range images. Similar to the previous work of Hegazy (2009) and Hegazy and Denzler (2009), we concentrate on the combination of color and texture information obtained from a standard CCD camera and range information from a time-of-flight camera. A beneficial combination of these two information sources with kernel-based methods requires an efficient method for hyperparameter optimization, which is available in Bayesian frameworks, such as Gaussian process classification (Section 2.6.1). As shown in Kapoor et al. (2010) and in the previous experiments and applications, regression with Gaussian process priors can be successfully integrated in an image categorization setting and is able to handle multiple kernels.

In the remainder of the current section, we briefly describe our sensor combination approach in Section 5.8.2. Details about the experimental dataset and the evaluation procedure is given in Section 5.8.3 and different aspects of the results are evaluated in Section 5.8.4.

5.8.2 Combining Multiple Sensor Information

We already described and explained nearly all parts of our approach in Chapter 2 and Chapter 4. Figure 5.27 gives a guideline how all parts are connected. As a ToF camera we use a Photonic Mixer Device (PMD Technologies 19k, Buxbaum (2002); Luan et al. (2001)), which additionally provides an intensity and an amplitude image. We do not use these images for our recognition system, because they suffer from severe noise artifacts. The range image also contains a lot of outliers and due to this reason some preprocessing is needed. We apply a 5×5 median filter, which has also shown to be helpful in a number of other ToF applications (Kähler et al., 2008; Böhme et al., 2010).

We extract several local features from both sensor images. In detail, we use OpponentSIFT features (Section 4.1.3) to represent the image of the CCD camera and three local range features (depth, surface normals, and curvature as explained in Section 4.1.4) to extract multiple characteristics of the preprocessed ToF range image. A summary of all local feature types used can be found in Table 5.3. Hegazy and Denzler (2009) use an interest detector, applied

Table 5.3: Overview of local feature types used in our approach.

Feature type	Sensor image	Histogram size	Reference
OpponentSIFT	CCD color	384	van de Sande et al. (2010)
Pixel depth	ToF range	64	Hetzel et al. (2001)
Surface normals	ToF range	8×8	Hetzel et al. (2001)
Shape index	ToF range	64	Hetzel et al. (2001)

to the ToF intensity image, to compute local descriptors only on corner-like image patches. However, as already mentioned, the intensity image is very noisy and due to the low resolution (160×120 pixels) provided by the ToF camera, this procedure often leads to uninformative interest points and a small number of local features. Therefore, we compute local features on a predefined grid as suggested by Hegazy (2009) and already done in our previous image categorization experiments.

Applying the spatial pyramid matching framework (Section 4.3.2) to each of the local feature sets results in different kernels K_i. We have one kernel function for color images and three kernels corresponding to ToF range images, which yields $R = 4$ kernels in total. Combining these kernels is done in a linear manner:

$$K(\boldsymbol{x}, \boldsymbol{x}') = \sum_{i=1}^{R} \exp(\eta_i) \, K_i(\boldsymbol{x}, \boldsymbol{x}') \ , \tag{5.8}$$

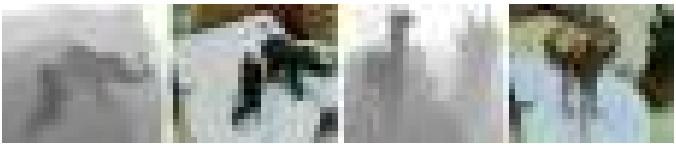
either with uniform weights $\eta_i = \frac{1}{R}$ or with weights $\boldsymbol{\eta}$ optimized by maximizing the joint marginal likelihood of a GP model for one-vs-all multi-class classification (Section 2.6.10).

5.8.3 Experimental Dataset and Setup

For our experiments, we utilize the dataset of Hegazy and Denzler (2009), which is up to now the only available object category database with color images from a CCD camera and range images from a ToF camera. It consists of a large set of 2d/2.5d image pairs[3] capturing different objects. Figure 5.28 shows

[3]Range images are often considered as 2.5d, because they only provide partial 3d information

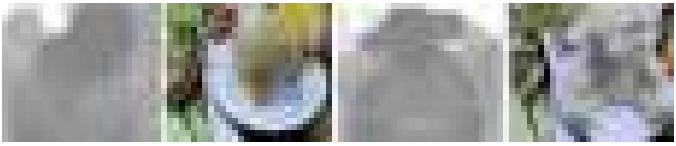
(a) *cars*



(b) *animals*

(c) *cups*

(d) *fruits*



(e) *toys*

Figure 5.28: Some examples of image pairs included in the dataset of Hegazy and Denzler (2009) with images obtained from a ToF camera and a visual sensor. The dataset includes high intra-class variabilities (*animals*, *fruits*, *toys*) and small interclass distance (*animals* and *toys*).

some example images, which help to judge the difficulty of the corresponding classification task. Images belong to five different generic object categories (*cars*, *toys*, *cups*, *fruits*, and *animals*) and each category consists of seven object instances (specific objects) with 32 image pairs. The dataset was collected and designed by Hegazy and Denzler (2009), such that it contains several common real-world challenges. Thus, images can contain multiple instances of the same category, are captured from different viewpoints and orientations, and include partial occlusions, truncations (*e.g.* due to image boundaries) as well as background clutter.

In contrast to previous work (Hegazy and Denzler, 2009), which use a predefined split into a training set with 100 images and a test set of 60 images for each category, we evaluate our approach using a varying number g of object instances for training ($32g$ image pairs) and the remaining images of the dataset for testing. As a performance measure we use the mean of the average recognition rate (Section 5.1.1) obtained from $Z = 50$ evaluations with a random selection of training instances.

5.8.4 Evaluation

In the following, we empirically support the following hypotheses:

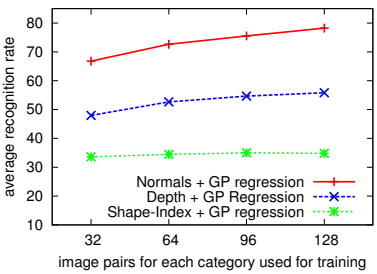
1. Our method significantly outperforms the previous approach of Hegazy (2009) (Section 5.8.4.4).

2. GP regression outperforms Laplace approximation (Section 5.8.4.4).

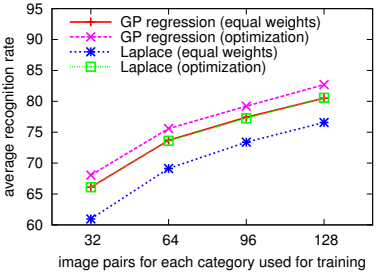
3. Generic object recognition, and especially visual categorization with few examples, benefits from range information
   (Section 5.8.4.2 and Section 5.8.4.3).

4. Combining range image kernels with GP likelihood optimization leads to better results than fixed equal kernel weights. However, weight optimization is not beneficial when all kernels and few training examples are used (Section 5.8.4.2).

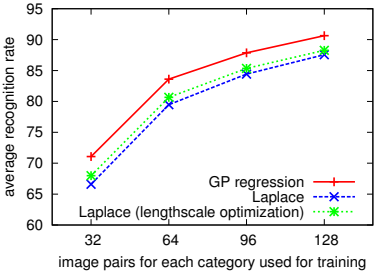
5. Local range features computed using surface normals lead to the best recognition performance among all other range features (Section 5.8.4.1).



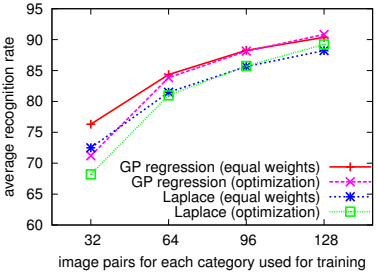
(a) Comparison of range feature types



(b) Combination of range features



(c) Visual features



(d) Combined features

Figure 5.29: Evaluation of (a) different types of range features; (b) GP methods with multiple range features; (c) GP methods with color features; (d) and combined features from the CCD camera and the ToF camera. 32 image pairs correspond to one object instance or type.

5.8.4.1 Evaluation of Range Feature Types

First of all, we compare the performance of all local range features presented in Section 4.1.4. Classification is done with GP regression and without additional hyperparameter optimization. The results are shown in Figure 5.29(a) for different numbers of object instances used for training. The performance ranking of the respective methods is clearly visible and local features calculated using surface normals result in the best average recognition rate. In an earlier work on scene recognition, we have already shown the ability of histograms of surface normals to provide discriminative features (Kemmler et al., 2009).

5.8.4.2 Evaluation of GP Classification and Kernel Combination

Let us have a look on the performance of GP regression compared to approximate GP classification with Laplace approximation (LA). Figure 5.29(c) shows the performance of both methods using the image kernel function of the CCD camera. GP regression significantly outperforms LA, which is a surprising result because of the theoretical suitability of LA for classification problems. We also test LA with hyperparameter optimization of the scaling factor σ_c included in the probit noise model and defined by Eq. (2.81) in Section 2.6.6. This additional hyperparameter optimization leads to a small performance gain, but is still inferior to GP regression.

Figure 5.29(b) shows that by combining multiple range kernels, the categorization performance increases compared to single range kernel functions. The best method is GP regression with weights optimized by marginal likelihood optimization as presented in Section 2.6.10. As observed in the previous experiment, LA does not lead to a performance gain, even with hyperparameter optimization.

The results for combined image kernels of both sensors are shown in Figure 5.29(d). Combining range data from the ToF sensor and images of the visual sensor leads to a superior categorization performance (76.3% using 1 instance) compared to the best results using a single sensor (71% with 1 instance). An interesting fact is that in this case, we do not benefit from weight optimization for kernel combination. This is likely due to overfitting in the presence of the highly discriminative color kernel. Therefore, one should prefer to use equal weights in those scenarios.

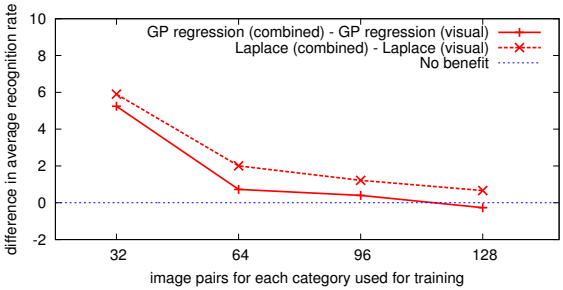


Figure 5.30: Performance benefit when using multiple sensors: Especially, learning with few examples benefits from multiple sensors. For GP regression the information present in 128 color training images seems to be sufficient, such that the noisy range images of the ToF camera do not help to improve the recognition performance anymore.

5.8.4.3 Learning with Few Examples Benefits from Multiple Sensors and Features

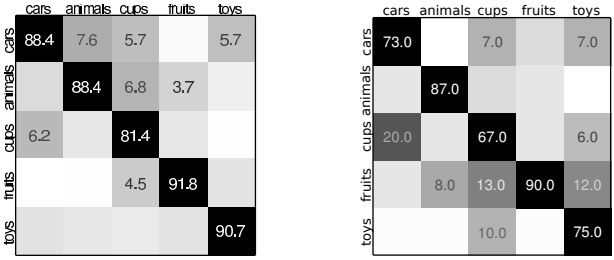
Figure 5.30 plots the performance gain (difference of average recognition rates) when using range features in addition to local features computed of the color image. It can be seen that the performance benefit due to sensor combination is most prevalent for few training examples. This result is intuitive, because in the case of multiple features from different sensors, more information is present in the small number of training examples. The observed benefit also arises when using multiple kernel functions and single sensors, *e.g.* standard color images and a combination of the PHoG kernel and other SPM kernels (Tommasi et al., 2010).

5.8.4.4 Comparison with Previous Work

We also compare our method with the previous state-of-the-art approach of Hegazy and Denzler (2009) and its extension using dense sampling (Hegazy, 2009). Note that these works are the only ones providing methods for ToF-based object categorization. In this experiment the same number of training examples is used to allow direct comparison of the recognition rates. The results are shown in Table 5.4. Our approach utilizing GP regression and hyperparameter optimization significantly outperforms their approach for single sensor data from

Table 5.4: Comparison of our GP-based approach to previous work with an equal number of training examples. (s) denotes range features computed on interest points only.

Reference	Method	Features	Recog. Rate
Hegazy and Denzler (2009)	Boosting	range features (s)	39.8 %
Hegazy (2009)	Boosting	range features	62.8 %
Our approach	GP Regression	range features	**79.2** %
Hegazy and Denzler (2009)	Boosting	combined features (s)	64.2 %
Hegazy (2009)	Boosting	combined features	78.4 %
Our approach	GP Regression	combined features	**88.1** %



(a) Our approach (b) Hegazy (2009)

Figure 5.31: Results of our GP approach and the method of Hegazy (2009) represented as confusion matrix. Only values above 3% are displayed and highlight difficult cases.

the ToF camera and combined information of both sensors. Even by using range features only, we achieve an average recognition rate of 79.2%, which is superior to the combined classification system of Hegazy (2009) with a recognition rate of 78.4%. The confusion matrix in Figure 5.31 highlights the still existing classification difficulties.

Chapter 6

Conclusions

The following chapter provides a summary of the results of this thesis. Furthermore, several research topics and open problems are mentioned, which were not covered by this thesis, but could be used as starting points for future research.

6.1 Summary and Thesis Contributions

The aim of this thesis was to develop and analyze algorithms and approaches that allow for learning visual object categories with few training examples. As elaborated in Section 1.1, small sample size problems are ill-posed without further incorporation of prior knowledge or restrictive assumptions. We studied the topic from the perspectives of transfer learning and one-class classification. Both paradigms were introduced in Chapter 1 along with an overview of previous work done in those areas. Chapter 2 provided insights into current machine learning approaches, such as kernel-based learning with Gaussian processes and support vector machines. Chapter 4 was dedicated to the computer vision aspects of this work concentrating on feature extraction and computing image-based kernel functions.

In Chapter 3, we presented our transfer learning approaches. Our transfer learning extensions of random decision forests are efficient and easy to integrate in current systems. The feature relevance method (Section 3.3) modifies the randomization procedure of random decision forests to concentrate on features that were considered as relevant in support classification tasks. Whereas, this

approach is restricted to binary transfer learning (Section 5.2), the regularized decision tree method (Section 3.2) allows for multi-class transfer learning scenarios (Section 5.4). The idea of this method is to re-estimate posterior probabilities corresponding to leaves in a decision tree by utilizing a prior from support classes and applying maximum a posteriori estimation. However, the main disadvantage of both methods is their inability to automatically determine suitable support tasks and the amount of information transferred. Our algorithm based on dependent Gaussian processes (Section 3.4) allows for non-parametric transfer learning with kernels. Another important property of our GP approach is its ability to adapt to different learning situations by automatically selecting support classification tasks and tuning the expected degree of similarity between the tasks. The selection procedure combines two opposed ideas: (1) pre-selecting support tasks by utilizing additional prior knowledge about semantic similarities (Section 3.4.4) and (2) choosing a task according to the expected performance gain estimated with the information present in all available training examples (Section 3.4.3). Our experiments in Section 5.3 showed the usefulness of this combination for visual recognition problems and the clear benefits in terms of recognition performance in comparison to previous work.

The other main topic tackled in this thesis is one-class classification. We showed how to utilize the Gaussian process framework for one-class classification and derived several different novelty scores. Our method is non-parametric, easy to implement and can be directly applied to visual out-of-vocabulary problems as studied in Section 5.5. Furthermore, we utilized this technique for detecting wire rope defects (Section 5.7), which is an application inherently suffering from the lack of training examples for defects. In general, our method is not restricted to visual applications and can be used for example for action recognition (Section 5.6) and detecting novel bacteria with Raman spectroscopy as shown in Kemmler et al. (2011). Our experiments in Section 5.5 and 5.7 showed that we outperform the support vector data description approach of Tax and Duin (2004), which is the current state-of-the-art technique in one-class classification. Additionally, we studied the effect of kernel hyperparameters on the resulting classification performance and outline the theoretical connections to previous approaches.

In Section 5.8, we analyzed the combination of a visual sensor and depth data from a time-of-flight camera for generic object recognition tasks. The information of the sensors is combined by multiple kernel learning with Gaussian processes. Our results validate that data from an additional depth sensor

increases the recognition performance, especially when confronted with few training examples. In contrast, optimizing weights for each sensor by maximizing the marginal likelihood derived from the GP framework only leads to an improvement of the recognition rate with a sufficient amount of training data. Our approach achieves significant higher recognition rates than previous Boosting approaches.

Apart from presenting the main aspects of the developed methods, an important goal of this thesis was also to analyze and describe the relations and connections of our methods to previous work. Each section in Chapter 3 contained further information on inherent assumptions and compared them to the ones used by already established algorithms in the same field. We believe that the derived connections offer important insights and the possibility to develop further extensions.

6.2 Future Work

Solving a problem always generates a large number of unsolved and interesting new research topics. A further property of research, which occurs especially in computer vision and machine learning, is that a problem is never completely solved but only up to a certain accuracy. Due to this reason, there is always space for improvement concerning the methods presented in this thesis. In the following, we give some ideas for future research directions.

Multi-class Transfer One of the areas that has been studied by this thesis. but only by a few other papers, is multi-class transfer. This transfer learning scenario is important in a real-world application where a robot has to discriminate between several object categories, but also transfers knowledge from previously learned categories to learn new visual concepts. An interesting topic for further research is to extend the dependent Gaussian process framework to this multi-class setting and to transfer without violating the discriminability. For example, in current realizations of multi-class GP classification, such as the model and algorithm presented in Rasmussen and Williams (2005, Section 3.5), the training examples of different classes are assumed to be independent. Modeling the dependencies between classes with their pairwise kernel functions requires an additional computational burden. However, it would allow incorporating prior knowledge about the similarity of target and support classes.

Multiple Task Correlation Parameters There is still a gap for possible improvement concerning binary transfer learning. For example, we selected a single support classification task from a given set and estimated a task correlation parameter. This restricts our method to transfer knowledge only from a single task at the same time. A more flexible option would be to apply the dependent Gaussian process framework directly to all given support tasks and estimate several task correlation parameters in a jointly manner. With J support tasks and one target task, two options are possible:

1. a full model that incorporates the dependencies between each task and therefore also between the support tasks ($J(J+1)/2$ parameters) and

2. a model that only utilizes the correlations of the target task and one of the support tasks (J parameters).

The still existing open problem of these procedures is that we need an efficient hyperparameter optimization method being able to handle multiple hyperparameters. With our greedy leave-one-out method this is not yet possible.

Part-based Transfer Learning An interesting topic for further research is part-based transfer learning. Instead of transferring all information present in the training examples, in some situations only some parts of a model are beneficial to transfer. This especially occurs in the visual domain in which very different features can be extracted, such as texture, color, object part constellations, and shape. A possible idea to perform part-based transfer learning is to use *automatic relevance determination (ARD)* and the dependent GP framework. The ARD approach uses a Gaussian kernel but with a different variance (or length-scale) for each element of the input vector (Rasmussen and Williams, 2005, Section 5.1). If the ARD kernel function is additionally modified such that the variances also dependent on the tasks of the two elements under consideration, the resulting dependent GP model would allow part-based knowledge transfer. However, this still requires an efficient hyperparameter optimization method as discussed previously.

Parameter Optimization for OCC In Section 5.5, we observed that another selection of the hyperparameter for one-class classification can lead to significant changes in the resulting classification performance. Due to this observation the question arises whether there is a possibility to automatically estimate optimal

hyperparameter values from training data. We already argued in Section 5.5.4 that this problem is ill-posed and related to the bandwidth selection problem of the Parzen estimator or the yet unsolved problem of selecting kernel hyperparameters for the SVDD method (Tax and Duin, 2004). Therefore, the problem seems to be inherent to OCC methods in general and whether there exists a generic parameter estimation method is an open question. However, if prior knowledge is available for a specific application or can be extracted from meta-data, the estimation of kernel hyperparameters might be possible.

Real-time Learning and Image Categorization A main disadvantage of Gaussian process based approaches is the cubic and linear runtime of learning and classification with respect to the training examples. Although our current implementation uses a GPU implementation of the Cholesky factorization (Gratton, 2008) to achieve faster learning times, the transfer learning method and our one-class classification algorithm are not able to handle more than ≈ 10.000 training examples on current hardware. However, there are several ideas how to further accelerate GP inference, such as the local approach of Urtasun and Darrell (2008) or our tree-based method presented in Fröhlich et al. (2011). Using these ideas to speed up our methods is an interesting topic for further research.

Multiple Kernels and Visual Features This thesis focused on machine learning aspects of transfer learning in the visual domain and we used standard features for image categorization, such as the concept of bag of visual words. However, there are a lot of extensions possible that are likely to lead to a large improvement of the absolute recognition performance. Instead of utilizing one specific feature type, the strategy of current state-of-the-art approaches is to combine multiple features and kernel functions, which extract different cues about an object category from the image (Kapoor et al., 2010). We analyzed these techniques in the context of generic object recognition with multiple sensors, but we did not investigated its use for transfer learning with dependent Gaussian processes.

Object Detection and Localization The experiments and applications in our work are image categorization tasks, *i.e.* we label an image instead of localizing object instances of different categories. Current object localization approaches

use a binary classifier applied to features computed in a sliding window. Due to the large number of classifier evaluations, this strategy requires an efficient classification method with a low computation time. A possible solution that would boost the performance of the kernel-based transfer learning method presented in this thesis, is to use kernel descriptors and formulate the approach as a linear algorithm (Bo et al., 2010; Gong and Lazebnik, 2011; Li et al., 2010).

Kernel Functions for Defect Detection in Wire Ropes Our approach for detecting defects in wire ropes does not exploit any structure knowledge of the rope. However, the work of Wacker and Denzler (2010) shows that by using a geometrical model of the rope, performances far beyond ours can be achieved. The method we developed allows using arbitrary kernel functions. One of the main advantages of kernels is that they allow incorporating prior knowledge. Therefore, kernels could be developed that directly assume a periodicity in the data.

Appendix A

Mathematical Details

The following sections list theorems and lemmata that have been partly used in Chapter 2 and Chapter 3.

A.1 The Representer Theorem

The representer theorem is one of the key results in the area of learning with kernels. It allows representing the decision function derived from a very general class of optimization functions by a weighted combination of kernel evaluations.

Theorem A.1 (Representer Theorem) *(Schölkopf and Smola, 2001)*
Let $\Omega : [0, \infty) \to \mathbb{R}$ be a strictly monotonically increasing function and $c : \left(\mathcal{X} \times \mathbb{R}^2\right)^n \to \mathbb{R} \cup \{\infty\}$ an arbitrary loss function. Then each minimizer $f \in \mathcal{H}$ of the regularized risk:

$$c((\boldsymbol{x}_1, y_1, f(\boldsymbol{x}_1)), \ldots, (\boldsymbol{x}_n, y_n, f(\boldsymbol{x}_n))) + \Omega(||f||_{\mathcal{H}}) \ , \tag{A.1}$$

can be written in the following form:

$$f(\boldsymbol{x}) = \sum_{i=1}^{n} \alpha_i K(\boldsymbol{x}_i, \boldsymbol{x}) \ . \tag{A.2}$$

Proof: A proof of this theorem can be found in Schölkopf and Smola (2001, Section 4.2, p. 90). □

A.2 Bounding the Standard Deviation of f

The following small theorem bounds the standard deviation of a decision function with terms depending on properties of the kernel function and the training set. It is used in Chapter 2 to illustrate the influence of several parameters.

Theorem A.2 (Bounding the Standard Deviation) *Let $\phi : \mathcal{X} \to \mathcal{H}$ be a feature map with corresponding kernel function K. If we are given a set of input examples $\mathbf{X} \subseteq \mathcal{X}$ and a function $f : \mathcal{X} \to \mathbb{R}$ such that the following representations hold*

$$f(\boldsymbol{x}) = \langle \boldsymbol{w}, \phi(\boldsymbol{x}) \rangle_{\mathcal{H}} + b \tag{A.3}$$

$$\boldsymbol{w} = \sum_{i=1}^{n} \alpha_i \phi(\boldsymbol{x}_i) \ , \tag{A.4}$$

then it is possible to give the following upper bound for the standard deviation of f:

$$\sigma_{\boldsymbol{x}}(f) \leq ||\boldsymbol{\alpha}|| \cdot \sqrt{\lambda_{max}(\mathbf{K})} \cdot \zeta_K \ , \tag{A.5}$$

with ζ_K depending only on the kernel function K and the distribution of inputs $\boldsymbol{x} \in \mathcal{X}$.

Proof:
We start by writing the standard deviation of f in terms of the norm of $\boldsymbol{w}$ in $\mathcal{H}$ and the standard deviation of transformed feature vectors $\phi(\boldsymbol{x}_i)$:

$$\sigma^2_{\boldsymbol{x}}(f(\boldsymbol{x})) = \int_{\mathcal{X}} \left(f(\boldsymbol{x}') - \mathbb{E}_{\boldsymbol{x}}(f(\boldsymbol{x}))\right)^2 p(\boldsymbol{x}') \, d\boldsymbol{x}' \tag{A.6}$$

$$= \int_{\mathcal{X}} \left(\langle \boldsymbol{w}, \phi(\boldsymbol{x}') - \mathbb{E}_{\boldsymbol{x}}(\phi(\boldsymbol{x})) \rangle_{\mathcal{H}}\right)^2 p(\boldsymbol{x}') \, d\boldsymbol{x}' \tag{A.7}$$

$$\leq \int_{\mathcal{X}} ||\boldsymbol{w}||^2_{\mathcal{H}} \cdot ||\phi(\boldsymbol{x}') - \mathbb{E}_{\boldsymbol{x}}(\phi(\boldsymbol{x}))||^2 \ p(\boldsymbol{x}') \, d\boldsymbol{x}' \tag{A.8}$$

$$\leq ||\boldsymbol{w}||^2_{\mathcal{H}} \cdot \Big(\int_{\mathcal{X}} K(\boldsymbol{x}', \boldsymbol{x}') p(\boldsymbol{x}') d\boldsymbol{x}' - \int_{\mathcal{X}\times\mathcal{X}} K(\boldsymbol{x}', \boldsymbol{x}) \, p(\boldsymbol{x}) \, p(\boldsymbol{x}') \, d\boldsymbol{x}' d\boldsymbol{x} \Big) \ . \tag{A.9}$$

The last factor of the bound only depends on the input distribution $p(\boldsymbol{x})$ and the kernel, therefore, we use the notation ζ_K^2 to refer to it. Due to the representation of $\boldsymbol{w}$, which only involves the set $\mathbf{X}$ (*cf.* representer theorem of Schölkopf and Smola (2001)), the norm of the hyperplane $\boldsymbol{w}$ is bounded by the maximum eigenvalue of the kernel matrix $\mathbf{K}$ and the norm of the coefficients $\boldsymbol{\alpha}$:

$$||\boldsymbol{w}||_{\mathcal{H}}^2 = \left|\left|\sum_{i=1}^{n} \alpha_i \phi(\boldsymbol{x}_i)\right|\right|_{\mathcal{H}}^2 = \sum_{i,j=1}^{n} \alpha_i \alpha_j \left\langle \phi(\boldsymbol{x}_i), \phi(\boldsymbol{x}_j) \right\rangle_{\mathcal{H}} \tag{A.10}$$

$$= \boldsymbol{\alpha}^T \mathbf{K} \boldsymbol{\alpha} \leq ||\boldsymbol{\alpha}||^2 \cdot \lambda_{\max}(\mathbf{K}) \ . \tag{A.11}$$

□

A.3 Blockwise Inversion and Schur Complement

We summarize and state some results about blockwise matrix factorization, which can be used for incremental learning with Gaussian processes and are also important to derive some properties of multivariate Gaussian distributions given in the next section.

Lemma A.3 (Blockwise Matrix Inversion) *Given a matrix* $\mathbf{M}$ *that is divided in the following block structure:*

$$\mathbf{M} = \begin{bmatrix} \mathbf{A} & \mathbf{B} \\ \mathbf{C} & \mathbf{D} \end{bmatrix} \ , \tag{A.12}$$

the inverse matrix of $\mathbf{M}$ *can be computed only using the inverse of* $\mathbf{A}$ *and the inverse Schur complement* $\mathbf{S} \stackrel{\text{def}}{=} -\mathbf{C}\mathbf{A}^{-1}\mathbf{B} + \mathbf{D}$*:*

$$\mathbf{M}^{-1} = \begin{bmatrix} \mathbf{A}^{-1} + \mathbf{A}^{-1}\mathbf{B}\mathbf{S}^{-1}\mathbf{C}\mathbf{A}^{-1} & -\mathbf{A}^{-1}\mathbf{B}\mathbf{S}^{-1} \\ -\mathbf{S}^{-1}\mathbf{C}\mathbf{A}^{-1} & \mathbf{S}^{-1} \end{bmatrix} \ . \tag{A.13}$$

Proof: Let us have a look at the following algebraic manipulations:

$$\begin{bmatrix} \mathbf{I} & \mathbf{0} \\ -\mathbf{C}\mathbf{A}^{-1} & \mathbf{I} \end{bmatrix} \cdot \mathbf{M} = \begin{bmatrix} \mathbf{A} & \mathbf{B} \\ \mathbf{0} & \mathbf{D} - \mathbf{C}\mathbf{A}^{-1}\mathbf{B} \end{bmatrix} = \begin{bmatrix} \mathbf{A} & \mathbf{B} \\ \mathbf{0} & \mathbf{S} \end{bmatrix} \tag{A.14}$$

$$\begin{bmatrix} \mathbf{A} & \mathbf{B} \\ \mathbf{0} & \mathbf{S} \end{bmatrix} \begin{bmatrix} \mathbf{I} & -\mathbf{A}^{-1}\mathbf{B} \\ \mathbf{0} & \mathbf{I} \end{bmatrix} = \begin{bmatrix} \mathbf{A} & \mathbf{0} \\ \mathbf{0} & \mathbf{S} \end{bmatrix} \ . \tag{A.15}$$

Note that the matrices used to generate the desired form of $\mathbf{M}$ are representing exactly the principle of Gaussian elimination. Finally, we have:

$$\begin{bmatrix} \mathbf{I} & \mathbf{0} \\ -\mathbf{C}\mathbf{A}^{-1} & \mathbf{I} \end{bmatrix} \cdot \mathbf{M} \cdot \begin{bmatrix} \mathbf{I} & -\mathbf{A}^{-1}\mathbf{B} \\ \mathbf{0} & \mathbf{I} \end{bmatrix} = \begin{bmatrix} \mathbf{A} & \mathbf{0} \\ \mathbf{0} & \mathbf{S} \end{bmatrix} . \tag{A.16}$$

The inverse matrices of the transformation and the diagonal block matrix on the right hand side are simple to calculate, due to their special structure. For this reason, we can easily compute the inverse of $\mathbf{M}$ using the Schur complement and the inverse matrix of the upper left submatrix:

$$\mathbf{M} = \begin{bmatrix} \mathbf{I} & \mathbf{0} \\ -\mathbf{C}\mathbf{A}^{-1} & \mathbf{I} \end{bmatrix}^{-1} \cdot \begin{bmatrix} \mathbf{A} & \mathbf{0} \\ \mathbf{0} & \mathbf{S} \end{bmatrix} \cdot \begin{bmatrix} \mathbf{I} & -\mathbf{A}^{-1}\mathbf{B} \\ \mathbf{0} & \mathbf{I} \end{bmatrix}^{-1} \tag{A.17}$$

$$\mathbf{M}^{-1} = \begin{bmatrix} \mathbf{I} & -\mathbf{A}^{-1}\mathbf{B} \\ \mathbf{0} & \mathbf{I} \end{bmatrix} \cdot \begin{bmatrix} \mathbf{A}^{-1} & \mathbf{0} \\ \mathbf{0} & \mathbf{S}^{-1} \end{bmatrix} \cdot \begin{bmatrix} \mathbf{I} & \mathbf{0} \\ -\mathbf{C}\mathbf{A}^{-1} & \mathbf{I} \end{bmatrix} \tag{A.18}$$

$$= \begin{bmatrix} \mathbf{A}^{-1} & -\mathbf{A}^{-1}\mathbf{B}\mathbf{S}^{-1} \\ \mathbf{0} & \mathbf{S}^{-1} \end{bmatrix} \begin{bmatrix} \mathbf{I} & \mathbf{0} \\ -\mathbf{C}\mathbf{A}^{-1} & \mathbf{I} \end{bmatrix} \tag{A.19}$$

$$= \begin{bmatrix} \mathbf{A}^{-1} + \mathbf{A}^{-1}\mathbf{B}\mathbf{S}^{-1}\mathbf{C}\mathbf{A}^{-1} & -\mathbf{A}^{-1}\mathbf{B}\mathbf{S}^{-1} \\ -\mathbf{S}^{-1}\mathbf{C}\mathbf{A}^{-1} & \mathbf{S}^{-1} \end{bmatrix} . \tag{A.20}$$

With a symmetric matrix $\mathbf{M}$, which can be written using $\mathbf{C} = \mathbf{B}^T$, we can further simplify the previous result:

$$\mathbf{M}^{-1} = \begin{bmatrix} \mathbf{A}^{-1} + \mathbf{A}^{-1}\mathbf{B}\mathbf{S}^{-1}\mathbf{B}^T\mathbf{A}^{-1} & -\mathbf{A}^{-1}\mathbf{B}\mathbf{S}^{-1} \\ \left(-\mathbf{A}^{-1}\mathbf{B}\mathbf{S}^{-1}\right)^T & \mathbf{S}^{-1} \end{bmatrix} . \tag{A.21}$$

□

Lemma A.4 (Matrix Inversion Lemma) *With suitably sized matrices and* $\mathbf{A}$ *being regular the following holds:*

$$\left(\mathbf{A} - \mathbf{B}\mathbf{D}^{-1}\mathbf{C}\right)^{-1} = \mathbf{A}^{-1} + \mathbf{A}^{-1}\mathbf{B}\left(\mathbf{D} - \mathbf{C}\mathbf{A}^{-1}\mathbf{B}\right)^{-1}\mathbf{C}\mathbf{A}^{-1} . \tag{A.22}$$

Proof: We follow the same argumentation as in the previous proof, but apply a different Gaussian elimination that yields:

$$\begin{bmatrix} \mathbf{I} & -\mathbf{B}\mathbf{D}^{-1} \\ \mathbf{0} & \mathbf{I} \end{bmatrix} \cdot \mathbf{M} \cdot \begin{bmatrix} \mathbf{I} & \mathbf{0} \\ -\mathbf{D}^{-1}\mathbf{C} & \mathbf{0} \end{bmatrix} \mathbf{I} = \begin{bmatrix} \tilde{\mathbf{S}} & \mathbf{0} \\ \mathbf{0} & \mathbf{D} \end{bmatrix} . \tag{A.23}$$

Using a similar derivation as done in the previous proof, we can state the blockwise inversion lemma with a modified inverse Schur complement $\tilde{\mathbf{S}} \stackrel{\text{def}}{=} -\mathbf{B}\mathbf{D}^{-1}\mathbf{C} + \mathbf{A}$ in the upper left corner of the matrix $\mathbf{M}^{-1}$. By comparing it to the original result of the blockwise inversion lemma, we immediately arrive at the statement of the current Lemma. □

A.4 Gaussian Identities

In the following, we present some fundamental facts about multivariate Gaussian distributions and their corresponding marginal and conditional distributions.

Lemma A.5 (Sum of Two Gaussians) *Let two multivariate Gaussian random variables $\boldsymbol{x}_1 \sim \mathcal{N}(\boldsymbol{\mu}_1, \mathbf{A}_1)$ and $\boldsymbol{x}_2 \sim \mathcal{N}(\boldsymbol{\mu}_2, \mathbf{A}_2)$ be given. It follows that the sum of both random variables is distributed according to:*

$$\boldsymbol{x}_1 + \boldsymbol{x}_2 \sim \mathcal{N}(\boldsymbol{\mu}_1 + \boldsymbol{\mu}_2, \mathbf{A}_1 + \mathbf{A}_2) \ . \tag{A.24}$$

Lemma A.6 (Conditional Gaussian Distribution) *Let $\boldsymbol{x} = (\boldsymbol{x}_a, \boldsymbol{x}_d)^T$ be a multivariate Gaussian random variable, i.e. $\boldsymbol{x} \sim \mathcal{N}(\boldsymbol{\mu}, \mathbf{M})$, with the following mean vector and covariance matrix*

$$\boldsymbol{\mu} = \begin{bmatrix} \boldsymbol{\mu}_a \\ \boldsymbol{\mu}_d \end{bmatrix} \qquad \mathbf{M} = \begin{bmatrix} \mathbf{A} & \mathbf{B}^T \\ \mathbf{B} & \mathbf{D} \end{bmatrix} \ . \tag{A.25}$$

It follows that the conditional distribution $p(\boldsymbol{x}_a \mid \boldsymbol{x}_d)$ is also a normal distribution with mean vector $\boldsymbol{\mu}'$ and covariance matrix $\mathbf{M}'$ (Petersen and Pedersen, 2008, Section 8.1.3):

$$\boldsymbol{\mu}' = \boldsymbol{\mu}_a + \mathbf{B}^T\mathbf{D}^{-1}(\boldsymbol{x}_d - \boldsymbol{\mu}_d) \qquad \mathbf{M}' = \mathbf{A} - \mathbf{B}^T\mathbf{D}^{-1}\mathbf{B} \ . \tag{A.26}$$

Proof: A proof can be found in Bishop (2006, Section 2.3.2, p. 89). □

Lemma A.7 (Maginal and Conditional Gaussian Distribution) *If two Gaussian random variables $\boldsymbol{x}_a$ and $\boldsymbol{x}_d$ are given and are distributed according to*

$$\boldsymbol{x}_a \sim \mathcal{N}(\boldsymbol{\mu}_a, \mathbf{A}) \tag{A.27}$$

$$\boldsymbol{x}_d | \boldsymbol{x}_a \sim \mathcal{N}(\mathbf{F}\boldsymbol{x}_a + \boldsymbol{b}, \mathbf{G}) \ , \tag{A.28}$$

then the marginal distribution of $\boldsymbol{x}_d$ is as follows:

$$\boldsymbol{x}_d \sim \mathcal{N}(\mathbf{F}\boldsymbol{\mu}_a + \boldsymbol{b}, \mathbf{G} + \mathbf{F}\mathbf{A}\mathbf{F}^T) \ . \tag{A.29}$$

Proof: A proof can be found in (Bishop, 2006, Section 2.3.3, p. 90-93). □

A.5 Kernel and Second Moment Matrix in Feature Space

We review some results given by Haasdonk and Pękalska (2010) highlighting the connection of the kernel matrix and the (sample) second moment matrix of the training data in feature space. We make use of the matrix $\mathbf{\Phi} = [\phi(\boldsymbol{x}_1), \ldots, \phi(\boldsymbol{x}_n]$ which contains all transformed training examples as columns. Furthermore, let the regularized kernel matrix be defined as $\mathbf{K}_{\text{reg}} = \mathbf{K} + \sigma_\varepsilon^2\mathbf{I}$ and the second moment matrix of the data in feature space defined as $\mathbf{C} = \frac{1}{n}\mathbf{\Phi}\mathbf{\Phi}^T$ with its regularized version $\mathbf{C}_{\text{reg}} = \frac{1}{n}\left(\mathbf{\Phi}\mathbf{\Phi}^T + \sigma_\varepsilon^2\mathbf{I}\right)$. Note that the regularization of the second moment matrix with $\sigma_\varepsilon^2 > 0$ is essential to ensure its regularity if the feature space is high-dimensional, which is the case for most common kernels. We can now state the following lemma:

Lemma A.8 (Second Moment Matrix in Feature Space) *Let n be the number of training examples, $\mathbf{C}_{reg}$ be the regularized second moment matrix of the training data in feature space and $\mathbf{K}_{reg}$ be the regularized kernel matrix as defined in the previous paragraph. As derived by Haasdonk and Pękalska (2010, Section 3.2), the following equation holds:*

$$\mathbf{\Phi}\mathbf{K}_{reg}^{-1} = \frac{1}{n}\mathbf{C}_{reg}^{-1}\mathbf{\Phi}. \tag{A.30}$$

Proof: The following basic identity establishes the first connection of $\mathbf{C}$ and $\mathbf{K}$:

$$\mathbf{\Phi}\mathbf{\Phi}^T\mathbf{\Phi} = n\mathbf{C}\mathbf{\Phi} = \mathbf{\Phi}\mathbf{K} \ . \tag{A.31}$$

This result also holds in a similar version for the regularized matrices:

$$n\mathbf{C}_{\text{reg}}\mathbf{\Phi} = n\mathbf{C}\mathbf{\Phi} + \sigma_\varepsilon^2\mathbf{\Phi} = \mathbf{\Phi}\mathbf{K} + \sigma_\varepsilon^2\mathbf{\Phi} = \mathbf{\Phi}\left(\mathbf{K} + \sigma_\varepsilon^2\mathbf{I}\right) = \mathbf{\Phi}\mathbf{K}_{\text{reg}} \ . \tag{A.32}$$

The regularity of $\mathbf{C}_{\text{reg}}$ and $\mathbf{K}_{\text{reg}}$ now allows us to multiply Eq. (A.32) with the corresponding inverse matrices, which leads to the final result. □

A.6 Details about Adapted LS-SVM

In the following, we reconsider the optimization problem of Adapted LS-SVM already reviewed in Section 3.4.6.4:

$$\begin{aligned} \underset{\boldsymbol{w}^{(\tau)}\in\mathbb{R}^D, b\in\mathbb{R}, \boldsymbol{\xi}\in\mathbb{R}^n}{\text{minimize}} \quad & \frac{1}{2}\left|\left|\boldsymbol{w}^{(\tau)} - \beta\boldsymbol{w}^{(s)}\right|\right|_{\mathcal{H}}^2 + \frac{C}{2}\sum_{i=1}^{n}\xi_i^2 \\ \text{subject to} \quad & \forall i = 1\ldots n:\ y_i = \left(\left\langle \boldsymbol{w}^{(\tau)}, \phi(\boldsymbol{x}_i)\right\rangle_{\mathcal{H}} + b\right) + \xi_i \ . \end{aligned} \tag{A.33}$$

The aim of this section is to derive expressions for the solution of the above optimization problem and especially its dual counterpart. We assume that the bias is incorporated in the kernel function and derive the Lagrangian L:

$$\begin{aligned} L(\boldsymbol{w}^{(\tau)}, \boldsymbol{\xi}, \boldsymbol{\alpha}) = & \frac{1}{2}\left|\left|\boldsymbol{w}^{(\tau)} - \beta\boldsymbol{w}^{(s)}\right|\right|_{\mathcal{H}}^2 + \frac{C}{2}\sum_{i=1}^{n}\xi_i^2 \\ & + \sum_{i=1}^{n}\alpha_i\left(y_i - \left\langle \boldsymbol{w}^{(\tau)}, \phi(\boldsymbol{x}_i)\right\rangle_{\mathcal{H}} - \xi_i\right) \ , \end{aligned} \tag{A.34}$$

with the following derivatives

$$\left\{\nabla_{\boldsymbol{w}^{(\tau)}} L\right\}(\boldsymbol{w}^{(\tau)}, b, \boldsymbol{\xi}, \boldsymbol{\alpha}) = \boldsymbol{w}^{(\tau)} - \beta\boldsymbol{w}^{(s)} - \sum_{i=1}^{n}\alpha_i\phi(\boldsymbol{x}_i) \tag{A.35}$$

$$\left\{\nabla_{\boldsymbol{\xi}} L\right\}(\boldsymbol{w}^{(\tau)}, b, \boldsymbol{\xi}, \boldsymbol{\alpha}) = C\boldsymbol{\xi} - \boldsymbol{\alpha} \ . \tag{A.36}$$

We now set both gradients to zero and incoporate the equations for $\boldsymbol{w}^{(\tau)}$ and $\boldsymbol{\xi}$ in L yielding the Lagrangian dual function $g(\boldsymbol{\alpha})$:

$$g(\boldsymbol{\alpha}) = \frac{1}{2}\left|\left|\sum_{i=1}^{n}\alpha_i\phi(\boldsymbol{x}_i)\right|\right|_{\mathcal{H}}^2 + \frac{1}{2C}\left|\left|\boldsymbol{\alpha}\right|\right|^2 + \boldsymbol{\alpha}^T\boldsymbol{z}_1 \ , \tag{A.37}$$

with the abbrevation $\boldsymbol{z}_1$:

$$\boldsymbol{z}_1 = \boldsymbol{y}_\tau - \boldsymbol{\xi} - \boldsymbol{\Phi}^T\boldsymbol{w}^{(\tau)} \tag{A.38}$$

$$= \boldsymbol{y}_\tau - \frac{1}{C}\boldsymbol{\alpha} - \boldsymbol{\Phi}^T\left(\beta\boldsymbol{w}^{(s)} + \boldsymbol{\Phi}\boldsymbol{\alpha}\right) \tag{A.39}$$

$$= \boldsymbol{y}_\tau - \frac{1}{C}\boldsymbol{\alpha} - \beta\boldsymbol{\Phi}^T\boldsymbol{w}^{(s)} - \mathbf{K}\boldsymbol{\alpha} \ . \tag{A.40}$$

Note that we concatenate the transformed examples to a single matrix $\mathbf{\Phi} = [\phi(\boldsymbol{x}_1), \ldots, \phi(\boldsymbol{x}_n)]$. We now use the set of support training examples $\mathbf{X}' = \{\boldsymbol{x}'_i\}_{i=1}^{n'}$ and write the normal vector of the support hyperplane as:

$$\boldsymbol{w}^{(s)} = \sum_{j=1}^{n'} \tilde{\alpha}_j \phi(\boldsymbol{x}'_j) = \mathbf{\Phi}' \boldsymbol{\alpha}' \ , \tag{A.41}$$

with $\mathbf{\Phi}' = [\phi(\boldsymbol{x}'_1), \ldots, \phi(\boldsymbol{x}'_{n'})]$. According to the representer theorem, this representation is valid and appropiate coefficients $\tilde{\alpha}_j$ exist. Multiplying $\mathbf{\Phi}$ with the hyperplane parameter yields the following kernel expressions:

$$\mathbf{\Phi}^T \boldsymbol{w}^{(s)} = \mathbf{\Phi}^T \mathbf{\Phi}' \boldsymbol{\alpha}' = \mathbf{K}_{\tau s} \, \boldsymbol{\alpha}' \ . \tag{A.42}$$

Including all derived expressions in the Lagrangian in Eq. (A.37) gives:

$$\begin{aligned} g(\boldsymbol{\alpha}) &= \frac{1}{2} \left|\left| \mathbf{\Phi} \boldsymbol{\alpha} \right|\right|^2_{\mathcal{H}} + \frac{1}{2C} \left|\left| \boldsymbol{\alpha} \right|\right|^2 + \boldsymbol{\alpha}^T \boldsymbol{y}_\tau - \frac{1}{C} \left|\left| \boldsymbol{\alpha} \right|\right|^2 - \beta \boldsymbol{\alpha}^T \mathbf{K}_{\tau s} \, \boldsymbol{\alpha}' - \boldsymbol{\alpha}^T \mathbf{K} \boldsymbol{\alpha} \\ &= \frac{1}{2} \boldsymbol{\alpha}^T \mathbf{K} \boldsymbol{\alpha} - \frac{1}{2C} \left|\left| \boldsymbol{\alpha} \right|\right|^2 + \boldsymbol{\alpha}^T \boldsymbol{y}_\tau - \beta \boldsymbol{\alpha}^T \mathbf{K}_{\tau s} \, \boldsymbol{\alpha}' - \boldsymbol{\alpha}^T \mathbf{K} \boldsymbol{\alpha} \\ &= -\frac{1}{2} \boldsymbol{\alpha}^T \left(\mathbf{K} + \frac{1}{C} \mathbf{I} \right) \boldsymbol{\alpha} + \boldsymbol{\alpha}^T \left(\boldsymbol{y}_\tau - \beta \mathbf{K}_{\tau s} \, \boldsymbol{\alpha}' \right) \ . \end{aligned} \tag{A.43}$$

Differentiating with respect to $\boldsymbol{\alpha}$ yields:

$$\left\{ \nabla g \right\} (\boldsymbol{\alpha}) = - \left(\mathbf{K} + \frac{1}{C} \mathbf{I} \right) \boldsymbol{\alpha} + \boldsymbol{y}_\tau - \beta \mathbf{K}_{\tau s} \, \boldsymbol{\alpha}' \ , \tag{A.44}$$

and finally we get the optimal vector $\boldsymbol{\alpha}$ for the target task, which was denoted as $\boldsymbol{\alpha}^{(\tau)}$ in Section 3.4.6.4, by setting the gradient to zero, including the equation for $\boldsymbol{\alpha}'$ and re-arranging:

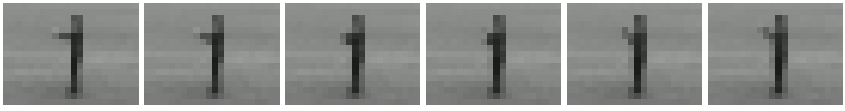
$$\boldsymbol{\alpha} = \left(\mathbf{K}_{\tau\tau} + \frac{1}{C} \mathbf{I} \right)^{-1} \left(\boldsymbol{y}_\tau - \beta \cdot \mathbf{K}_{\tau s} \cdot \left(\mathbf{K}_{ss} + \frac{1}{C} \mathbf{I} \right)^{-1} \boldsymbol{y}_s \right) \ . \tag{A.45}$$

Appendix B

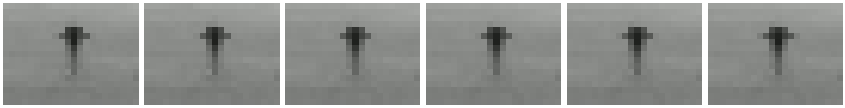
Experimental Details

Table B.1: Mapping Caltech-101 category names to WordNet terms.

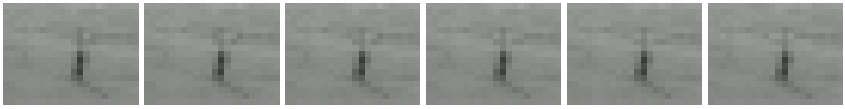
Caltech term	WordNet term	WN sense	Caltech term	WordNet term	WN sense
airplanes	airplane	1	barrel	barrel	2
bass	bass	8	beaver	beaver	7
binocular	binoculars	1	buddha	-	-
car_side	car	1	ceiling_fan	fan	1
cougar_body	cougar	1	cougar_face	cougar	1
crayfish	crayfish	4	crocodile_head	crocodile	1
dalmatian	dalmatian	2	dolphin	dolphin	2
emu	emu	2	faces	face	1
faces_easy	face	-	flamingo_head	flamingo	1
garfield	-	-	inline_skate	skate	1
lamp	lamp	2	leopards	leopard	2
lobster	lobster	2	motorbikes	motorbike	1
nautilus	nautilus	2	octopus	octopus	2
pyramid	-	-	schooner	schooner	2
scorpion	scorpion	3	sea_horse	sea_horse	2
snoopy	-	-	stop_sign	sign	2
tick	tick	2	water_lilly	water_lily	1
wild_cat	wildcat	3	wrench	wrench	3
yin_yang	symbol	1			



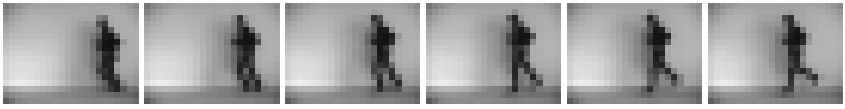
(a) *boxing*



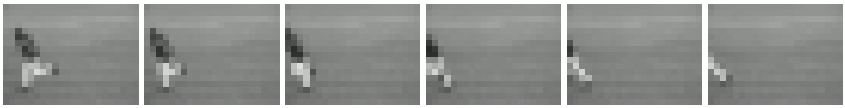
(b) *hand-clapping*



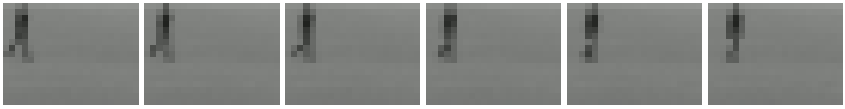
(c) *hand-waving*



(d) *jogging*



(e) *running*



(f) *walking*

Figure B.1: Example images of the KTH action dataset, which comprises six different action categories.

Table B.2: Standard parameter values used to learn random decision forests in our experiments. The influence of the parameters on the learning process is described in Section 2.3.2. † This depth was never reached in practice and can be regarded as disabling the termination criterion. ‡ Termination only if all examples in the current node belong to the same category or if there are not more than ξ_{n} examples left.

Description	Notation	Value
sampled features	$\lvert\mathcal{R}\rvert$	300
sampled splits	$\lvert Q\rvert$	15
number of trees	T	200
termination criterion maximum depth	ξ_{d}	$100^{\dagger}$
termination criterion impurity	$\xi_{\mathcal{J}}$	$0.0^{\ddagger}$
termination criterion examples	ξ_{n}	10

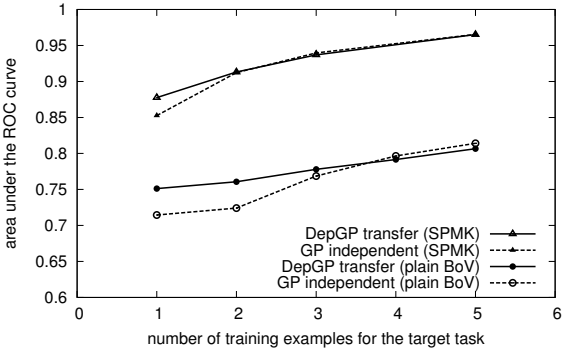
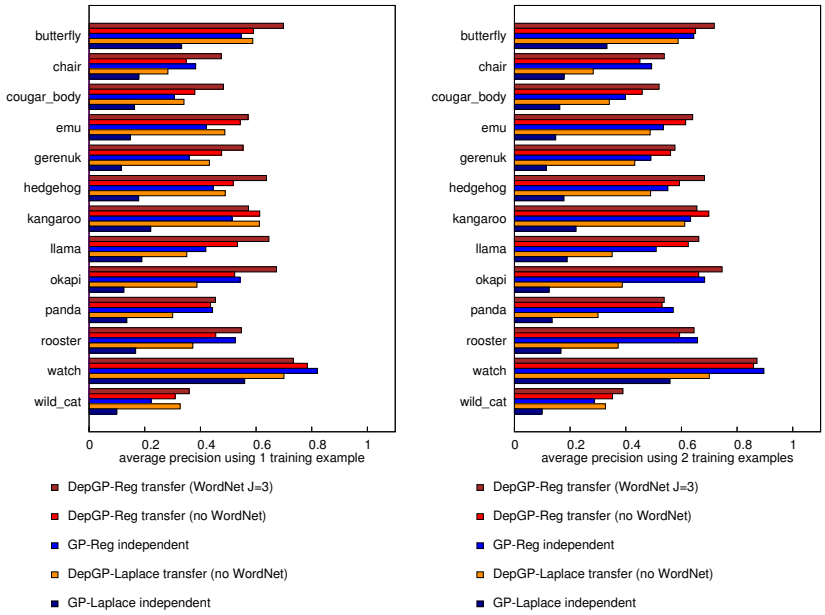


Figure B.2: Comparison of all BoV Histograms and Spatial Pyramid Matching: The graph compares plain BoV histograms as used in Section 5.2 and the results of spatial pyramid matching utilized in Section 5.3. Performance values are given for binary transfer learning with the target task *okapi* and the support task *gerenuk* (*cf.* Figure 5.5). It can be seen that the SPMK framework has a significant benefit with respect to the recognition performance and compared to plain BoV histograms. Another interesting observation is that the performance gain of transfer learning is more prominent for the weaker feature representation of standard BoV histograms. This can be explained by the fact that SPMK introduces more prior knowledge about the task and transfer learning is likely to be more beneficial if the independent learning performance is low.



(a) 1 training example

(b) 2 training examples

	butterfly	*chair*	*cougar_body*	*emu*	*gerenuk*
WordNet $J = 3$	1e-15	1e-15	1e-15	1e-15	1e-15
no WordNet	1e-7	1e-3	1e-8	1e-13	1e-12
	hedgehog	*kangaroo*	*llama*	*okapi*	
WordNet $J = 3$	1e-15	1e-6	1e-15	1e-12	
no WordNet	1e-7	1e-11	1e-12	*n.s.*	
	panda	*rooster*	*watch*	*wild_cat*	
WordNet $J = 3$	*n.s.*	*n.s.*	1e-13	1e-15	
no WordNet	*n.s.*	1e-5	1e-4	1e-13	

(c) Are the results in Figure B.3(a) significant compared to independent learning? The p-value of the paired t-Test is below ... (*n.s.*= not significant)

Figure B.3: Detailed Results of Heterogeneous Transfer Learning: (a,b) Caltech-101 results for our transfer learning approach with and without pre-selection of support classification tasks using WordNet and for independent learning using a single or two training examples. (c) Results of a significance test.

Bibliography

Ryan Prescott Adams, Iain Murray, and David MacKay. The Gaussian Process Density Sampler. In *Advances in Neural Information Processing Systems*, pages 9–16. 2009. (Cited on pages 20, 111 and 112)

Sameer Agarwal, Noah Snavely, Ian Simon, Steven M. Seitz, and Richard Szeliski. Building Rome in a Day. In *Proceedings of the 2009 International Conference on Computer Vision (ICCV'09)*, pages 72 – 79. 2009. doi:10.1109/ICCV.2009.5459148. (Cited on page 116)

Ferit Akova, Murat Dundar, V. Jo Davisson, E. Daniel Hirleman, Arun K. Bhunia, J. Paul Robinson, and Bartek Rajwa. A machine-learning approach to detecting unknown bacterial serovars. *Stat. Anal. Data Min.*, 3:289–301, October 2010. doi:10.1002/sam.v3:5. (Cited on page 20)

Yonatan Amit, Michael Fink, Nathan Srebro, and Shimon Ullman. Uncovering shared structures in multiclass classification. In *Proceedings of the 2007 international conference on Machine learning (ICML'07)*, pages 17–24. ACM Press, New York, NY, USA, 2007. doi:10.1145/1273496.1273499. (Cited on pages 14, 15 and 81)

Rie Kubota Ando and Tong Zhang. A Framework for Learning Predictive Structures from Multiple Tasks and Unlabeled Data. *Journal of Machine Learning Research*, 6:1817–1853, 2005. (Cited on page 13)

Mykhaylo Andriluka, Lorenz Weizsäcker, and Thomas Hofmann. Multi-class Classification with Dependent Gaussian Processes. In *Proceedings of 12th International Conference on Applied Stochastic Models and Data Analysis*. Crete, Greece, May 2007. (Cited on pages 100 and 101)

Andreas Argyriou, Theodoros Evgeniou, and Massimiliano Pontil. Multi-Task Feature Learning. In Bernhard Schölkopf, John C. Platt, and Thomas Hoffman, editors, *Advances in Neural Information Processing Systems*, pages 41–48. MIT Press, 2006. (Cited on pages 14, 17 and 81)

Andreas Argyriou, Andreas Maurer, and Massimiliano Pontil. An Algorithm for Transfer Learning in a Heterogeneous Environment. In *ECML PKDD '08: Proceedings of the 2008 European Conference on Machine Learning and Knowledge Discovery in Databases - Part I*, pages 71–85. Springer-Verlag, Berlin, Heidelberg, 2008. doi:10.1007/978-3-540-87479-9_23. (Cited on page 17)

Ferid Bajramovic. *Kernel-basierte Objektverfolgung*. Master's thesis, Universität Passau, 2004. (Cited on page 69)

Evgeniy Bart and Shimon Ullman. Cross-Generalization: Learning Novel Classes from a Single Example by Feature Replacement. In *Proceedings of the 2005 IEEE Conference on Computer Vision and Pattern Recognition (CVPR'05)*, pages 672–679. 2005. doi:10.1109/CVPR.2005.117. (Cited on page 155)

Frédéric Bastien, Yoshua Bengio, Arnaud Bergeron, Nicolas Boulanger-Lewandowski, Thomas Breuel, Youssouf Chherawala, Moustapha Cisse, Myriam Côté, Dumitru Erhan, Jeremy Eustache, Xavier Glorot, Xavier Muller, Sylvain Pannetier Lebeuf, Razvan Pascanu, Salah Rifai, Francois Savard, and Guillaume Sicard. Deep Self-Taught Learning for Handwritten Character Recognition. *arXiv*, abs/1009.3589, 2010. (Cited on page 3)

Sabri Bayoudh, Harold Mouchère, Laurent Miclet, and Éric Anquetil. Learning a Classifier with Very Few Examples: Analogy Based and Knowledge Based Generation of New Examples for Character Recognition. In *Proceedings of the 2007 European conference on Machine learning (ECML'07)*, pages 527–534. 2007. (Cited on page 11)

Vaishak Belle, Thomas Deselaers, and Stefan Schiffer. Randomized Trees for Real-Time One-Step Face Detection and Recognition. In *International Conference on Pattern Recognition 2008*, pages 1 – 4. Tampa, Florida, USA, 2008. doi:10.1109/ICPR.2008.4761365. (Cited on page 82)

Shai Ben-David and Michael Lindenbaum. Localization vs. Identification of Semi-Algebraic Sets. *Machine Learning*, 32:207–224, 1998. doi:10.1023/A:1007447530834. (Cited on page 6)

J. Bennett and S. Lanning. The netflix prize. In *Proceedings of KDD Cup and Workshop*, volume 2007. Citeseer, 2007. (Cited on page 3)

D. Beymer and T. Poggio. Face recognition from one example view. In *Proceedings of the 1995 International Conference on Computer Vision (ICCV'95)*, pages 500 – 507. 1995. doi:10.1109/ICCV.1995.466898. (Cited on page 11)

I. Biederman. Recognition-by-components: a theory of human image understanding. *Psychol Rev*, 94(2):115–147, Apr 1987. (Cited on page 1)

Christopher M. Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer, 1 edition, 2006. ISBN 0387310738. (Cited on pages 5, 18, 20, 25, 31, 32, 42, 44, 46, 49, 50, 60, 103, 105, 148, 197 and 198)

J. Blasco, N. Aleixos, and E. Molto. Computer vision detection of peel defects in citrus by means of a region oriented segmentation algorithm. *Journal of Food Engineering*, 81(3):535 – 543, 2007. doi:10.1016/j.jfoodeng.2006.12.007. (Cited on page 168)

Liefeng Bo, Xiaofeng Ren, and Dieter Fox. Kernel Descriptors for Visual Recognition. In J. Lafferty, C. K. I. Williams, J. Shawe-Taylor, R.S. Zemel, and A. Culotta, editors, *Advances in Neural Information Processing Systems*, pages 244–252. 2010. (Cited on page 192)

Liefeng Bo and Cristian Sminchisescu. Efficient Match Kernel between Sets of Features for Visual Recognition. In Y. Bengio, D. Schuurmans, J. Lafferty, C. K. I. Williams, and A. Culotta, editors, *Advances in Neural Information Processing Systems*, pages 135–143. 2009. (Cited on page 128)

Paul Bodesheim. *Object Discovery - Unüberwachtes Lernen von Objektkategorien*. Master's thesis, University of Jena, 2011. (Cited on pages 128 and 129)

Martin Böhme, Martin Haker, Thomas Martinetz, and Erhardt Barth. Shading constraint improves accuracy of time-of-flight measurements. *Computer Vision and Image Understanding*, 114:1329–1335, December 2010. doi:10.1016/j.cviu.2010.08.001. (Cited on page 178)

Edwin Bonilla, Kian Ming Chai, and Chris Williams. Multi-task Gaussian Process Prediction. In *Advances in Neural Information Processing Systems*, pages 153–160. MIT Press, 2008. (Cited on pages 15, 89, 100 and 101)

E.V. Bonilla, F.V. Agakov, and C.K.I. Williams. Kernel multi-task learning using task-specific features. In *Proceedings of the 11th AISTATS*. 2007. (Cited on page 15)

Anna Bosch, Andrew Zisserman, and Xavier Munoz. Representing shape with a spatial pyramid kernel. In *Proceedings of the 2007 ACM international conference on Image and video retrieval (CIVR'07)*, pages 401–408. ACM, New York, NY, USA, 2007. doi:10.1145/1282280.1282340. (Cited on pages 128, 129 and 157)

Anna Bosch, Andrew Zisserman, and Xavier Munoz. Scene Classification Using a Hybrid Generative/Discriminative Approach. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(4):712–727, 2008. doi:10.1109/TPAMI.2007.70716. (Cited on pages 22, 115 and 122)

Stephen Boyd and Lieven Vandenberghe. *Convex Optimization*. Cambridge University Press, New York, NY, USA, 2004. ISBN 0521833787. (Cited on page 47)

Pavel Brazdil, Christophe Giraud-Carrier, Carlos Soares, and Ricardo Vilalta. *Metalearning - Applications to Data Mining*. Springer, 2009. (Cited on page 11)

Leo Breiman. Bagging predictors. *Machine Learning*, 24(2):123–140, 1996. doi:10.1023/A:1018054314350. (Cited on pages 22 and 32)

Leo Breiman. Random Forests. *Machine Learning*, 45(1):5–32, October 2001. doi:10.1023/A:1010933404324. (Cited on pages 36, 37 and 85)

Leo Breiman, Jerome Friedman, Charles J. Stone, and R. A. Olshen. *Classification and Regression Trees*. Chapman and Hall/CRC, 1 edition, January 1984. ISBN 0412048418. (Cited on pages 34 and 36)

Ann L. Brown and Mary Jo Kane. Preschool Children Can Learn to Transfer: Learning to Learn and Learning from Example. *Cognitive Psychology*, 20:493–523, 1988. (Cited on page 7)

Bernd Buxbaum. *Optische Laufzeitentfernungsmessung und CDMA auf Basis der PMD-Technologie mittels phasenvariabler PN-Modulation*. Shaker Verlag, 2002. (Cited on page 178)

Bin Cao, Sinno Jialin Pan, Yu Zhang, Dit-Yan Yeung, and Qiang Yang. Adaptive Transfer Learning. In *Proceedings of the 24th AAAI Conference on Artificial Intelligence (AAAI-10)*. 2010. (Cited on pages 101 and 149)

Kian Ming Chai. Generalization Errors and Learning Curves for Regression with Multi-task Gaussian Processes. In *Advances in Neural Information Processing Systems*, pages 279–287. 2009. (Cited on pages 89, 90, 94 and 101)

Kian Ming Adam Chai, Christopher K. I. Williams, Stefan Klanke, and Sethu Vijayakumar. Multi-task Gaussian Process Learning of Robot Inverse Dynamics. In *Advances in Neural Information Processing Systems*, pages 265–272. MIT Press, 2008. (Cited on pages 15 and 100)

Yunqiang Chen, Xiang Zhou, and Thomas S. Huang. One-Class SVM for Learning in Image Retrieval. In *Proceedings of the IEEE Conference on Image Processing*. 2001. doi:10.1109/ICIP.2001.958946. (Cited on pages 19 and 105)

Roland T. Chin and Charles A. Harlow. Automated Visual Inspection: A Survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 4(6):557–573, 1982. doi:10.1109/TPAMI.1982.4767309. (Cited on pages 3 and 168)

Young-Sik Choi. Least squares one-class support vector machine. *Pattern Recognition Letters*, 30(13):1236–1240, 2009. doi:10.1016/j.patrec.2009.05.007. (Cited on page 113)

Koby Crammer and Yoram Singer. Learning Algorithms for Enclosing Points in Bregmanian Spheres. In Bernhard Schölkopf and Manfred K. Warmuth, editors, *Learning Theory and Kernel Machines*, volume 2777 of *Lecture Notes in Computer Science*, pages 388–402. Springer Berlin / Heidelberg, 2003. doi:10.1007/978-3-540-45167-9_29. (Cited on page 113)

Nello Cristianini and John Shawe-Taylor. *An Introduction to Support Vector Machines and Other Kernel-based Learning Methods*. Cambridge University Press, March 2000. ISBN 0521780195. (Cited on pages 5 and 6)

Gabriela Csurka and Florent Perronnin. A Simple High Performance Approach to Semantic Segmentation. In *Proceedings of the 2008 British Machine Vision Conference (BMVC'08)*, pages 213–222. 2008. (Cited on pages 24 and 123)

Felipe Cucker and Steve Smale. On the mathematical foundations of learning. *Bulletin of the American Mathematical Society*, 39:1–49, 2002. doi:10.1.1.129.1553. (Cited on page 5)

Yan Cui, Sebastian Schuon, Chan Derek, Sebastian Thrun, and Christian Theobalt. 3D Shape Scanning with a Time-of-Flight Camera. In *Proceedings of the 2010 IEEE Conference on Computer Vision and Pattern Recognition (CVPR'10)*, pages 1173 – 1180. 2010. doi:10.1109/CVPR.2010.5540082. (Cited on page 176)

Navneet Dalal and Bill Triggs. Histograms of Oriented Gradients for Human Detection. In *Proceedings of the 2005 IEEE Conference on Computer Vision and Pattern Recognition (CVPR'05)*, pages 886–893. 2005. doi:10.1109/CVPR.2005.177. (Cited on pages 43, 128 and 170)

Chris Dance, Jutta Willamowski, Lixin Fan, Cedric Bray, and Gabriela Csurka. Visual categorization with bags of keypoints. In *ECCV International Workshop on Statistical Learning in Computer Vision*. 2004. (Cited on pages 121 and 122)

Jesse Davis and Mark Goadrich. The Relationship Between Precision-Recall and ROC Curves. In *Proceedings of the 2006 International Conference on Machine Learning (ICML'06)*, pages 233–240. 2006. doi:10.1145/1143844.1143874. (Cited on page 134)

J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. ImageNet: A Large-Scale Hierarchical Image Database. In *Proceedings of the 2009 IEEE Conference on Computer Vision and Pattern Recognition (CVPR'09)*, pages 248 – 255. 2009. doi:10.1109/CVPR.2009.5206848. (Cited on pages 2 and 137)

Joachim Denzler. *Probabilistische Zustandsschätzung und Aktionsauswahl im Rechnersehen*. Logos Verlag Berlin, 2003. (Cited on page 31)

Thomas Deselaers and Vittorio Ferrari. Visual and Semantic Similarity in ImageNet. In *Proceedings of the 2011 IEEE Conference on Computer Vision and Pattern Recognition (CVPR'11)*. 2011. (Cited on pages 98 and 150)

Thomas Deselaers, Daniel Keysers, and Hermann Ney. Improving a Discriminative Approach to Object Recognition Using Image Patches. In *Proceedings of the 2005 Annual Symposium of the German Association for Pattern Recognition (DAGM'05)*, pages 326–333. 2005. (Cited on page 122)

Gy. Dorkó and C. Schmid. Selection of Scale-Invariant Parts for Object Class Recognition. In *Proceedings of the 2003 International Conference on Computer Vision (ICCV'03)*, pages 634 – 639. 2003. (Cited on page 122)

Richard O. Duda, Peter E. Hart, and David G. Stork. *Pattern Classification (2nd Edition)*. Wiley-Interscience, November 2000. ISBN 0471056693. (Cited on page 11)

Dan E. Dudgeon and Russell M. Mersereau. *Multidimensional Digital Signal Processing*. Prentice-Hall, 1984. (Cited on page 117)

M. Murat Dundar, E. Daniel Hirleman, Arun K. Bhunia, J. Paul Robinson, and Bartek Rajwa. Learning with a non-exhaustive training dataset: a case study: detection of bacteria cultures using optical-scattering technology. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '09, pages 279–288. ACM, New York, NY, USA, 2009. doi:10.1145/1557019.1557055. (Cited on pages 19 and 20)

Boris Epshtein and Shimon Ullman. Identifying Semantically Equivalent Object Fragments. In *Proceedings of the 2005 IEEE Conference on Computer Vision and Pattern Recognition (CVPR'05)*, pages 2–9. 2005. doi:10.1109/CVPR.2005.180. (Cited on page 122)

Boris Epshtein and Shimon Ullman. Satellite Features for the Classification of Visually Similar Classes. In *Proceedings of the 2006 IEEE Conference on Computer Vision and Pattern Recognition (CVPR'06)*, pages 2079–2086. 2006. doi:10.1109/CVPR.2006.262. (Cited on page 122)

Mark Everingham, Luc Gool, Christopher K. Williams, John Winn, and Andrew Zisserman. The Pascal Visual Object Classes (VOC) Challenge. *International Journal of Computer Vision*, 88:303–338, June 2010. doi:10.1007/s11263-009-0275-4. (Cited on pages 1, 100 and 137)

Theodoros Evgeniou, Charles A. Micchelli, and Massimiliano Pontil. Learning Multiple Tasks with Kernel Methods. *Journal of Machine Learning Research*, 6:615–637, 2005. (Cited on pages 14 and 15)

A. Farhadi, I. Endres, D. Hoiem, and D. Forsyth. Describing objects by their attributes. In *Proceedings of the 2009 IEEE Conference on Computer Vision and Pattern Recognition (CVPR'09)*, pages 1778 – 1785. 2009. doi:10.1109/CVPR.2009.5206772. (Cited on pages 16 and 43)

Ali Farhadi, Ian Endres, and Derek Hoiem. Attribute-Centric Recognition for Cross-category Generalization. In *Proceedings of the 2010 IEEE Conference on Computer Vision and Pattern Recognition (CVPR'10)*, pages 2352 – 2359. 2010. doi:10.1109/CVPR.2010.5539924. (Cited on page 16)

Li Fei-Fei. Knowledge transfer in learning to recognize visual objects classes. In *Proceedings of the International Conference on Development and Learning (ICDL)*. 2006. (Cited on page 11)

Li Fei-Fei, Rob Fergus, and Pietro Perona. One-Shot Learning of Object Categories. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(4):594–611, 2006. doi:10.1109/TPAMI.2006.79. (Cited on pages 7, 13, 79, 98, 135, 136, 137, 138, 140, 155 and 157)

P. Felzenszwalb, D. Mcallester, and D. Ramanan. A Discriminatively Trained, Multiscale, Deformable Part Model. In *Proceedings of the 2008 IEEE Conference on Computer Vision and Pattern Recognition (CVPR'08)*, pages 1 – 8. 2008. doi:10.1109/CVPR.2008.4587597. (Cited on pages 2 and 43)

Andras Ferencz, Erik G. Learned-Miller, and Jitendra Malik. Learning to Locate Informative Features for Visual Identification. *International Journal of Computer Vision*, 77(1-3):3–24, 2008. doi:10.1007/s11263-007-0093-5. (Cited on page 12)

R. Fergus, P. Perona, and A. Zisserman. Object class recognition by unsupervised scale-invariant learning. In *Proceedings of the 2003 IEEE Conference on Computer Vision and Pattern Recognition (CVPR'03)*, volume 2, pages 264–271. 2003. (Cited on pages 14 and 116)

Rob Fergus, Yair Weiss, and Antonio Torralba. Semi-Supervised Learning in Gigantic Image Collections. In *Advances in Neural Information Processing Systems*, pages 522–530. 2009. (Cited on page 6)

Michael Fink. Object Classification from a Single Example Utilizing Class Relevance Pseudo-Metrics. In *Advances in Neural Information Processing*

Systems, volume 17, pages 449–456. The MIT Press, 2004. (Cited on pages 12, 18, 135, 136, 152 and 153)

Michael Fink and Pietro Perona. Mutual Boosting for Contextual Inference. In Sebastian Thrun, Lawrence K. Saul, and Bernhard Schölkopf, editors, *Advances in Neural Information Processing Systems*. MIT Press, 2003. (Cited on page 16)

Michael Fink, Shai Shalev-Shwartz, Yoram Singer, and Shimon Ullman. Online multiclass learning by interclass hypothesis sharing. In *Proceedings of the 2006 International Conference on Machine Learning (ICML'06)*, pages 313–320. ACM Press, New York, NY, USA, 2006. doi:10.1145/1143844.1143884. (Cited on page 13)

Michael Fink and Shimon Ullman. From Aardvark to Zorro: A Benchmark for Mammal Image Classification. *International Journal of Computer Vision*, 77(1-3):143–156, 2008. doi:10.1007/s11263-007-0066-8. (Cited on page 137)

Y. Freund and R.E. Schapire. A decision–theoretic generalization of on–line learning and an application to boosting. *Journal of Computer and System Sciences*, 55(1):119–139, 1997. (Cited on page 13)

Jerome Friedman, Trevor Hastie, and Robert Tibshirani. Additive Logistic Regression: A Statistical View of Boosting. *The Annals of Statistics*, 28(2):337–374, 2000. (Cited on pages 6 and 31)

Björn Fröhlich, Erik Rodner, and Joachim Denzler. A Fast Approach for Pixelwise Labeling of Facade Images. In *Proceedings of the 2010 International Conference on Pattern Recognition (ICPR'10)*, volume 7, pages 3029–3032. 2010. (Cited on pages 22, 23 and 124)

Björn Fröhlich, Erik Rodner, Michael Kemmler, and Joachim Denzler. Efficient Gaussian process classification using random decision forests. *Pattern Recognition and Image Analysis*, 21:184–187, 2011. doi:10.1134/S1054661811020337. (Cited on pages 24 and 191)

Carolina Galleguillos, Boris Babenko, Andrew Rabinovich, and Serge J. Belongie. Weakly Supervised Object Localization with Stable Segmentations. In *Proceedings of the 2008 European Conference on Computer Vision (ECCV'08)*, pages 193–207. 2008. (Cited on page 17)

V. Ganapathi, C. Plagemann, D. Koller, and S. Thrun. Real Time Motion Capture using a Single Time-Of-Flight Camera. In *Proceedings of the 2010 IEEE Conference on Computer Vision and Pattern Recognition (CVPR'10)*, pages 755 – 762. 2010. doi:10.1109/CVPR.2010.5540141. (Cited on page 176)

Pierre Geurts, Damien Ernst, and Louis Wehenkel. Extremely randomized trees. *Maching Learning*, 63(1):3–42, 2006. doi:10.1007/s10994-006-6226-1. (Cited on pages 37, 85 and 86)

Yunchao Gong and Svetlana Lazebnik. Comparing Data-Dependent and Data-Independent Embeddings for Classification and Ranking of Internet Images. In *Proceedings of the 2011 IEEE Conference on Computer Vision and Pattern Recognition (CVPR'11)*. 2011. (Cited on page 192)

Steven Gratton. Cholesky factorization in CUDA. `http://www.ast.cam.ac.uk/~stg20/cuda/cholesky/`, 2008. (Cited on page 191)

K. Grauman and T. Darrell. The pyramid match kernel: discriminative classification with sets of image features. In *Proceedings of the 2005 International Conference on Computer Vision (ICCV'05)*, volume 2, pages 1458–1465. 2005. doi:10.1109/ICCV.2005.239. (Cited on pages 125 and 127)

Kristen Grauman and Trevor Darrell. The Pyramid Match Kernel: Efficient Learning with Sets of Features. *Journal of Machine Learning Research*, 8:725–760, 2007. (Cited on pages 43, 125 and 127)

Kristen Grauman, Kristen Grauman, Trevor Darrell, and Trevor Darrell. Approximate correspondences in high dimensions. In *Advances in Neural Information Processing Systems*, volume 2006, pages 505–512. 2006. (Cited on page 125)

G. Griffin, A. Holub, and P. Perona. Caltech-256 object category dataset. Technical Report UCB/CSD-04-1366, California Institute of Technology, 2007. (Cited on pages 4 and 135)

Isabelle Guyon, Steve Gunn, Masoud Nikravesh, and Lotfi A. Zadeh. *Feature Extraction: Foundations and Applications (Studies in Fuzziness and Soft Computing)*. Springer, 2006. ISBN 3540354875. (Cited on pages 82 and 84)

Bernard Haasdonk and Elżbieta Pękalska. Classification with Kernel Mahalanobis Distance Classifiers. In Andreas Fink, Berthold Lausen, Wilfried

Seidel, and Alfred Ultsch, editors, *Advances in Data Analysis, Data Handling and Business Intelligence*, Studies in Classification, Data Analysis, and Knowledge Organization, pages 351–361. Springer, 2010. doi:10.1007/978-3-642-01044-6_32. (Cited on pages 110 and 198)

Daniel Haase, Esther-Sabrina Wacker, Ernst Günter Schukat-Talamazzini, and Joachim Denzler. Analysis of Structural Dependencies for the Automatic Visual Inspection of Wire Ropes. In *Proceedings of the 15th International Workshop on Vision, Modelling, and Visualization (VMV)*, pages 49–56. 2010. (Cited on page 169)

Martin Haker, Martin Böhme, Thomas Martinetz, and Erhardt Barth. Scale-Invariant Range Features for Time-of-Flight Camera Applications. In *CVPR 2008 Workshop on Time-of-Flight-based Computer Vision (TOF-CV)*, pages 1–6. 2008. (Cited on page 178)

D. Han, L. Bo, and C. Sminchisescu. Selection and Context for Action Recognition. In *Proceedings of the 2009 International Conference on Computer Vision (ICCV'09)*, pages 1933 – 1940. 2009. doi:10.1109/ICCV.2009.5459427. (Cited on page 165)

Junfeng He, Shih-Fu Chang, and Lexing Xie. Fast Kernel Learning for Spatial Pyramid Matching. In *Proceedings of the 2008 IEEE Conference on Computer Vision and Pattern Recognition (CVPR'08)*. 2008. doi:10.1109/CVPR.2008.4587636. (Cited on page 128)

Doaa Hegazy. *Boosting for Generic 2D/3D Object Recognition*. Ph.D. thesis, Friedrich Schiller University of Jena, 2009. (Cited on pages 120, 177, 178, 179, 181, 184 and 185)

Doaa Hegazy and Joachim Denzler. Combining Appearance and Range Based Information for Multi-class Generic Object Recognition. In *CIARP '09: Proceedings of the 14th Iberoamerican Conference on Pattern Recognition*, pages 741–748. Springer-Verlag, Berlin, Heidelberg, 2009. doi:10.1007/978-3-642-10268-4_87. (Cited on pages 178, 179, 180, 181, 184 and 185)

Guenter Hetzel, Bastian Leibe, Paul Levi, and Bernt Schiele. 3D Object Recognition from Range Images using Local Feature Histograms. In *Proceedings of the 2001 IEEE Conference on Computer Vision and Pattern Recognition (CVPR'01)*, volume 2, pages 394–399. 2001. (Cited on pages 120, 177 and 179)

Tin Kam Ho. The Random Subspace Method for Constructing Decision Forests. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(8):832–844, 1998. doi:10.1109/34.709601. (Cited on page 37)

V. Hodge and J. Austin. A Survey of Outlier Detection Methodologies. *Artificial Intelligence Review*, 22(2):85–126, 2004. (Cited on page 18)

Jennifer Hoeting, David Madigan, Adrian Raftery, and Chris Volinsky. Bayesian Model Averaging. *Statistical Science*, 14:382–401, 1999. (Cited on page 30)

Heiko Hoffmann. Kernel PCA for novelty detection. *Pattern Recognition*, 40:863–874, March 2007. doi:10.1016/j.patcog.2006.07.009. (Cited on page 20)

Thomas Hofmann. Unsupervised Learning by Probabilistic Latent Semantic Analysis. *Machine Learning*, 42(1-2):177–196, 2001. (Cited on page 22)

D. Hoiem, C. Rother, and J. Winn. 3D LayoutCRF for Multi-View Object Class Recognition and Segmentation. In *Proceedings of the 2007 IEEE Conference on Computer Vision and Pattern Recognition (CVPR'07)*, pages 1–8. 2007. doi:10.1109/CVPR.2007.383045. (Cited on page 75)

Derek Hoiem, Alexei A. Efros, and Martial Hebert. Geometric Context from a Single Image. In *Proceedings of the 2005 International Conference on Computer Vision (ICCV'05)*, volume 1, pages 654 – 661. IEEE, October 2005. (Cited on page 16)

Eva Hörster, Rainer Lienhart, and Malcolm Slaney. Image retrieval on large-scale image databases. In *Proceedings of the 2007 ACM international conference on Image and video retrieval (CIVR'07)*, pages 17–24. ACM, New York, NY, USA, 2007. doi:10.1145/1282280.1282283. (Cited on page 124)

Gary B. Huang, Vidit Jain, and Erik G. Learned-Miller. Unsupervised Joint Alignment of Complex Images. In *Proceedings of the 2007 International Conference on Computer Vision (ICCV'07)*, pages 1–8. IEEE, 2007. doi:10.1109/ICCV.2007.4408858. (Cited on page 12)

Vidit Jain and Erik Learned-Miller. Online Domain Adaptation of a Pre-Trained Cascade of Classifiers. In *Proceedings of the 2011 IEEE Conference on Computer Vision and Pattern Recognition (CVPR'11)*. 2011. (Cited on page 12)

Wei Jiang, Shih-Fu Chang, and Alexander C. Loui. Kernel Sharing With Joint Boosting For Multi-Class Concept Detection. In *Proceedings of the 2007 IEEE Conference on Computer Vision and Pattern Recognition (CVPR'07)*, pages 1–8. 2007. doi:10.1109/CVPR.2007.383483. (Cited on page 13)

George H. John, Ron Kohavi, and Karl Pfleger. Irrelevant Features and the Subset Selection Problem. In *International Conference on Machine Learning*, pages 121–129. 1994. (Cited on pages 83 and 84)

Frederic Jurie and Bill Triggs. Creating Efficient Codebooks for Visual Recognition. In *Proceedings of the 2005 International Conference on Computer Vision (ICCV'05)*, pages 604–610. 2005. doi:10.1109/ICCV.2005.66. (Cited on page 123)

Ashish Kapoor, Kristen Grauman, Raquel Urtasun, and Trevor Darrell. Gaussian Processes for Object Categorization. *International Journal of Computer Vision*, 88(2):169–188, 2010. doi:10.1007/s11263-009-0268-3. (Cited on pages 15, 66, 107, 176, 177, 178 and 191)

Samuel Kaski and Jaakko Peltonen. Learning from Relevant Tasks Only. In *Proceedings of the 2007 European conference on Machine learning (ECML'07)*, pages 608–615. Springer-Verlag, Berlin, Heidelberg, 2007. doi:10.1007/978-3-540-74958-5_59. (Cited on page 17)

Michael Kemmler and Joachim Denzler. Selection of Relevant Features for Raman Spectroscopy Using Supervised Classification Techniques. In *Chemometrics in Analytical Chemistry (CAC)*. 2010. Conference presentation only. (Cited on page 87)

Michael Kemmler, Erik Rodner, and Joachim Denzler. Global Context Extraction for Object Recognition Using a Combination of Range and Visual Features. In *Proceedings of the Dynamic 3D Imaging Workshop*, pages 96–109. 2009. (Cited on pages 22, 120 and 183)

Michael Kemmler, Erik Rodner, and Joachim Denzler. One-Class Classification with Gaussian Processes. In *Proceedings of the Asian Conference on Computer Vision*, volume 2, pages 489–500. 2010. (Cited on page 104)

Michael Kemmler, Erik Rodner, and Joachim Denzler. Automatic Identification of Novel Bacteria using Raman Spectroscopy and Gaussian Processes. *Journal of Raman Spectroscopy*, 2011. Submitted. (Cited on page 188)

J. Kiefer. Sequential Minimax Search for a Maximum. *Proceedings of the American Mathematical Society*, 4(3):502–506, 1953. (Cited on page 96)

Hyun-Chul Kim and Jaewook Lee. Pseudo-density Estimation for Clustering with Gaussian Processes. In *Advances in Neural Networks - ISNN*, volume 3971, pages 1238–1243. 2006. (Cited on pages 107 and 111)

Alexander Kläser, Marcin Marszałek, and Cordelia Schmid. A Spatio-Temporal Descriptor Based on 3D-Gradients. In *Proceedings of the 2008 British Machine Vision Conference (BMVC'08)*, pages 995–1004. sep 2008. (Cited on page 165)

Jan J. Koenderink and Andrea J. van Doorn. Surface shape and curvature scales. *Image and Vision Computing*, 10(8):557–564, October 1992. doi:10.1016/0262-8856(92)90076-F. (Cited on page 120)

Harold W. Kuhn. The Hungarian Method for the Assignment Problem. In *50 Years of Integer Programming 1958-2008*, pages 29–47. Springer Berlin Heidelberg, 2010. ISBN 978-3-540-68279-0. doi:10.1007/978-3-540-68279-0_2. (Cited on page 125)

A. Kumar. Computer-Vision-Based Fabric Defect Detection: A Survey. *IEEE Transactions on Industrial Electronics*, 55(1):348 –363, jan. 2008. doi:10.1109/TIE.1930.896476. (Cited on page 167)

A. Kumar and G.K.H. Pang. Defect detection in textured materials using optimized filters. *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, 32(5):553 – 570, oct 2002. doi:TSMCB.2002.1033176. (Cited on page 168)

Ludmila I. Kuncheva and Juan José Rodríguez. An Experimental Study on Rotation Forest Ensembles. In Michal Haindl, Josef Kittler, and Fabio Roli, editors, *Multiple Classifier Systems*, volume 4472 of *Lecture Notes in Computer Science*, pages 459–468. Springer, 2007. (Cited on page 37)

Olaf Kähler, Erik Rodner, and Joachim Denzler. On Fusion of Range and Intensity Information Using Graph-Cut for Planar Patch Segmentation. *International Journal of Intelligent Systems Technologies and Applications*, 5(3/4):365–373, 2008. (Cited on page 178)

Carmen Lai, David M. J. Tax, Robert P. W. Duin, Elzbieta Pekalska, and Pavel Paclík. On Combining One-Class Classifiers for Image Database Retrieval. In *MCS '02: Proceedings of the Third International Workshop on Multiple Classifier Systems*, pages 212–221. Springer-Verlag, London, UK, 2002. (Cited on pages 19 and 105)

Christoph H. Lampert and Oliver Krömer. Weakly-Paired Maximum Covariance Analysis for Multimodal Dimensionality Reduction and Transfer Learning. In *Proceedings of the 2010 European Conference on Computer Vision (ECCV'10)*, pages 566–579. 2010. doi:10.1007/978-3-642-15552-9_41. (Cited on page 16)

Christoph H. Lampert, Hannes Nickisch, and Stefan Harmeling. Learning To Detect Unseen Object Classes by Between-Class Attribute Transfer. In *Proceedings of the 2009 IEEE Conference on Computer Vision and Pattern Recognition (CVPR'09)*, pages 951–958. 2009. doi:10.1109/CVPR.2009.5206594. (Cited on page 15)

Robert Lange. *3D Time-of-Flight Distance Measurement with Custom Solid-State Image Sensors in CMOS/CCD-Technology*. Ph.D. thesis, University of Siegen, 2000. (Cited on page 176)

Ivan Laptev. On Space-Time Interest Points. *International Journal of Computer Vision*, 64(2-3):107–123, 2005. doi:10.1007/s11263-005-1838-7. (Cited on pages 163 and 164)

Ivan Laptev, Marcin Marszalek, Cordelia Schmid, and Benjamin Rozenfeld. Learning realistic human actions from movies. In *Proceedings of the 2008 IEEE Conference on Computer Vision and Pattern Recognition (CVPR'08)*, pages 1–8. 2008. doi:10.1109/CVPR.2008.4587756. (Cited on pages 163 and 165)

H. Larochelle, D. Erhan, and Y. Bengio. Zero-data learning of new tasks. In *AAAI Conference on Artificial Intelligence*. 2008. (Cited on page 15)

Neil D. Lawrence. Probabilistic non-linear principal component analysis with Gaussian process latent variable models. *Journal of Machine Learning Research*, 6:1783–1816, 2005. (Cited on pages 13, 101 and 112)

Neil D. Lawrence, John C. Platt, and Michael I. Jordan. Extensions of the Informative Vector Machine. In *Deterministic and Statistical Methods in Machine Learning*, pages 56–87. 2004. (Cited on pages 14, 68, 99 and 101)

Svetlana Lazebnik, Cordelia Schmid, and Jean Ponce. Semi-local Affine Parts for Object Recognition. In *Proceedings of the 2004 British Machine Vision Conference (BMVC'04)*, volume 2, pages 779–788. 2004. (Cited on pages 135, 136, 153 and 154)

Svetlana Lazebnik, Cordelia Schmid, and Jean Ponce. Beyond Bags of Features: Spatial Pyramid Matching for Recognizing Natural Scene Categories. In *Proceedings of the 2006 IEEE Conference on Computer Vision and Pattern Recognition (CVPR'06)*, pages 2169–2178. 2006a. doi:10.1109/CVPR.2006.68. (Cited on pages 121, 124, 125, 127 and 178)

Svetlana Lazebnik, Cordelia Schmid, and Jean Ponce. A Discriminative Framework for Texture and Object Recognition Using Local Image Features. In *Toward Category-Level Object Recognition*, pages 423–442. Springer, 2006b. (Cited on pages 136, 137, 153 and 154)

Erik G. Learned-Miller. Data Driven Image Models through Continuous Joint Alignment. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(2):236–250, 2006. doi:10.1109/TPAMI.2006.34. (Cited on page 11)

Jun Won Lee and C. Giraud-Carrier. Transfer Learning in Decision Trees. In *International Joint Conference on Neural Networks (IJCNN)*, pages 726–731. Aug. 2007. doi:10.1109/IJCNN.2007.4371047. (Cited on page 81)

Su-In Lee, Vassil Chatalbashev, David Vickrey, and Daphne Koller. Learning a meta-level prior for feature relevance from multiple related tasks. In *Proceedings of the 2007 international conference on Machine learning (ICML'07)*, pages 489–496. 2007. doi:10.1145/1273496.1273558. (Cited on pages 8, 81 and 88)

Vincent Lepetit, Pascal Lagger, and Pascal Fua. Randomized Trees for Real-Time Keypoint Recognition. In *Proceedings of the 2005 IEEE Conference on Computer Vision and Pattern Recognition (CVPR'05)*, pages 775–781. 2005. doi:10.1109/CVPR.2005.288. (Cited on page 75)

Kobi Levi, Michael Fink, and Yair Weiss. Learning From a Small Number of Training Examples by Exploiting Object Categories. In *Proceedings of the 2004 Conference on Computer Vision and Pattern Recognition Workshop (CVPRW'04)*, volume 6, pages 96–104. 2004. (Cited on pages 13 and 88)

Kobi Levi and Yair Weiss. Learning Object Detection from a Small Number of Examples: the Importance of Good Features. In *Proceedings of the 2004 IEEE Conference on Computer Vision and Pattern Recognition (CVPR'04)*, volume 2, pages 53–60. June 2004. (Cited on page 11)

Fuxin Li, Catalin Ionescu, and Cristian Sminchisescu. Random Fourier approximations for skewed multiplicative histogram kernels. In *Proceedings of the 2010 Annual Symposium of the German Association for Pattern Recognition (DAGM'10)*, pages 262–271. Springer-Verlag, Berlin, Heidelberg, 2010. (Cited on pages 49 and 192)

X.J. Li and I. Guskov. 3D object recognition from range images using pyramid matching. In *International Conference on Computer Vision (ICCV'07)*, pages 1–6. 2007. doi:10.1109/ICCV.2007.4408829. (Cited on page 178)

Tsz-Wai Rachel Lo and J. Paul Siebert. Local feature extraction and matching on range images: 2.5D SIFT. *Computer Vision and Image Understanding*, 113(12):1235–1250, 2009. doi:10.1016/j.cviu.2009.06.005. (Cited on page 177)

David G. Lowe. Distinctive Image Features from Scale-Invariant Keypoints. *International Journal of Computer Vision*, 60(2):91–110, 2004. doi:10.1023/B:VISI.0000029664.99615.94. (Cited on pages 116, 118, 119 and 121)

Xuming Luan, Rudolf Schwarte, Zhigang Zhang, Zhanping Xu, Horst-Guenther Heinol, Bernd Buxbaum, Thorsten Ringbeck, and H. Hess. Three-dimensional intelligent sensing based on the PMD technology. In *Proceedings of SPIE 4540*, page 482. 2001. doi:10.1117/12.450695. (Cited on page 178)

Marcin Marszalek and Cordelia Schmid. Spatial Weighting for Bag-of-Features. In *Proceedings of the 2006 IEEE Conference on Computer Vision and Pattern Recognition (CVPR'06)*, pages 2118–2125. 2006. doi:10.1109/CVPR.2006.288. (Cited on page 122)

Marcin Marszalek and Cordelia Schmid. Semantic Hierarchies for Visual Object Recognition. In *Proceedings of the 2007 IEEE Conference on Computer Vision and Pattern Recognition (CVPR'07)*, pages 1–7. 2007. doi:10.1109/CVPR.2007.383272. (Cited on page 97)

Arman Melkumyan and Fabio Ramos. Multi-Kernel Gaussian Processes. In *Proceedings of the NIPS Workshop on Learning from Multiple Sources with Applications to Robotics*. 2009. (Cited on page 101)

J. Mercer. Functions of positive and negative type, and their connection with the theory of integral equations. *Philosophical Transactions of the Royal Society, London*, 209:415–446, 1909. (Cited on page 41)

Ingo Mierswa and Michael Wurst. Efficient Case Based Feature Construction for Heterogeneous Learning Tasks. In *Proceedings of the 2005 European conference on Machine learning (ECML'05)*, pages 641–648. 2005. (Cited on page 17)

K Mikolajczyk, T Tuytelaars, C Schmid, A Zisserman, J Matas, F Schaffalitzky, T Kadir, and L van Gool. A comparison of affine region detectors. *International Journal of Computer Vision*, 65(7):43–72, November 2005. doi:10.1007/s11263-005-3848-x. (Cited on page 117)

Krystian Mikolajczyk and Cordelia Schmid. A performance evaluation of local descriptors. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(10):1615–1630, 2005. doi:10.1109/TPAMI.2005.188. (Cited on page 119)

J. Milgram, Mohamed Cheriet, and R. Sabourin. Estimating accurate multi-class probabilities with support vector machines. In *International Joint Conference on Neural Networks (IJCNN'05)*, volume 3, pages 1906 – 1911 vol. 3. july-4 aug. 2005. doi:10.1109/IJCNN.2005.1556171. (Cited on page 66)

Erik G. Miller, Nicholas E. Matsakis, and Paul A. Viola. Learning from One Example through Shared Densities on Transforms. In *Proceedings of the 2000 IEEE Conference on Computer Vision and Pattern Recognition (CVPR'00)*, pages 464–471. 2000. doi:10.1109/CVPR.2000.855856. (Cited on pages 11, 18 and 74)

T. Minka and R. Picard. Learning how to learn is learning with point sets. 1997. Unpublished manuscript. (Cited on page 30)

Thomas P. Minka. *A family of algorithms for approximate bayesian inference*. Ph.D. thesis, Massachusetts Institute of Technology, 2001. AAI0803033. (Cited on pages 61, 104 and 107)

Thomas P. Minka. Bayesian model averaging is not model combination. Technical report, MIT, 2002. (Cited on pages 31 and 32)

D. Moll. Innovative procedure for visual rope inspection. *Lift Report*, 29(3):10–14, 2003. (Cited on pages 167, 169 and 170)

Florent Monay, Pedro Quelhas, Daniel Gatica-Perez, and Jean-Marc Odobez. Constructing Visual Models with a Latent Space Approach. In *Subspace, Latent Structure and Feature Selection*, pages 115–126. 2005. (Cited on page 122)

Frank Moosmann, Diane Larlus, and Frederic Jurie. Learning Saliency Maps for Object Categorization. In *ECCV International Workshop on The Representation and Use of Prior Knowledge in Vision*. 2006a. (Cited on page 124)

Frank Moosmann, Bill Triggs, and Frédéric Jurie. Fast Discriminative Visual Codebooks using Randomized Clustering Forests. In *Advances in Neural Information Processing Systems*, pages 985–992. 2006b. (Cited on pages 123 and 164)

H. Nickisch and C. E. Rasmussen. Approximations for Binary Gaussian Process Classification. *Journal of Machine Learning Research*, 9:2035–2078, 10 2008. (Cited on page 60)

Hannes Nickisch and Carl Edward Rasmussen. Gaussian Mixture Modeling with Gaussian Process Latent Variable Models. In *Proceedings of the 2010 Annual Symposium of the German Association for Pattern Recognition (DAGM'10)*, pages 272–282. 2010. (Cited on page 112)

Juan Carlos Niebles, Hongcheng Wang, and Li Fei-Fei. Unsupervised Learning of Human Action Categories Using Spatial-Temporal Words. *International Journal of Computer Vision*, 79(3):299–318, 2008. doi:10.1007/s11263-007-0122-4. (Cited on pages 121, 163 and 165)

Heinrich Niemann. *Pattern Analysis and Understanding*. Springer Series in Information Sciences, 2 edition, 1990. (Cited on page 4)

Eric Nowak and Frederic Jurie. Learning Visual Similarity Measures for Comparing Never Seen Objects. In *Proceedings of the 2007 IEEE Conference on Computer Vision and Pattern Recognition (CVPR'07)*, pages 1–8. 2007. (Cited on page 81)

Eric Nowak, Frederic Jurie, and Bill Triggs. Sampling Strategies for Bag-of-Features Image Classification. In *Proceedings of the 2006 European Conference on Computer Vision (ECCV'06)*, pages 490–503. 2006. (Cited on pages 117, 122 and 123)

Terence Odlin. *Language transfer: Cross-linguistic influence in language learning*. Cambridge University Press, 1989. ISBN 0521378095. (Cited on page 7)

Sangmin Oh, Anthony Hoogs, Amitha Perera, Naresh Cuntoor, Chia-Chih Chen, Jong Taek Lee, Saurajit Mukherjee, J. K. Aggarwal, Hyungtae Lee, Larry Davis, Eran Swears, Xioyang Wang, Qiang Ji, Kishore Reddy, Mubarak Shah, Carl Vondrick, Hamed Pirsiavash, Deva Ramanan, Jenny Yuen, Antonio Torralba, Bi Song, Anesco Fong, Amit Roy-Chowdhury, and Mita Desai. A Large-scale Benchmark Dataset for Event Recognition in Surveillance Video. In *Proceedings of the 2011 IEEE Conference on Computer Vision and Pattern Recognition (CVPR'11)*. 2011. (Cited on page 163)

Björn Ommer and Joachim M. Buhmann. Learning the Compositional Nature of Visual Objects. In *Proceedings of the 2007 IEEE Conference on Computer Vision and Pattern Recognition (CVPR'07)*, pages 1–8. 2007. doi:10.1109/CVPR.2007.383154. (Cited on page 122)

Francesco Orabona, Claudio Castellini, Barbara Caputo, Angelo Emanuele Fiorilla, and Giulio Sandini. Model adaptation with least-squares SVM for adaptive hand prosthetics. In *Proceedings of the 2009 IEEE international conference on Robotics and Automation*, ICRA'09, pages 439–445. IEEE Press, Piscataway, NJ, USA, 2009. (Cited on page 102)

Mark Palatucci, Dean Pomerleau, Geoffrey Hinton, and Tom Mitchell. Zero-shot Learning with Semantic Output Codes. In *Advances in Neural Information Processing Systems*, pages 1410–1418. 2009. (Cited on page 16)

Sinno Jialin Pan and Qiang Yang. A Survey on Transfer Learning. *IEEE Transactions on Knowledge and Data Engineering*, 22(10):1345–1359, October 2010. doi:10.1109/TKDE.2009.191. (Cited on pages 7, 10 and 11)

D. Parikh and C.L. Zitnick. The Role of Features, Algorithms and Data in Visual Recognition. In *Proceedings of the 2010 IEEE Conference on Computer Vision and Pattern Recognition (CVPR'10)*, pages 2328–2335. 2010. doi:10.1109/CVPR.2010.5539920. (Cited on page 1)

Ted Pedersen, Siddharth Patwardhan, and Jason Michelizzi. WordNet: : Similarity - Measuring the Relatedness of Concepts. In *Proceedings of Fifth Annual*

Meeting of the North American Chapter of the Association for Computational Linguistics (NAACL-2004), pages 38–41. 2004. (Cited on pages 16, 21, 89 and 97)

Kaare Brandt Petersen and Michael Syskind Pedersen. *Matrix Cookbook*. 2008. (Cited on pages 68 and 197)

Gianluigi Pillonetto, Francesco Dinuzzo, and Giuseppe De Nicolao. Bayesian Online Multitask Learning of Gaussian Processes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2:193–205, February 2010. doi:10.1109/TPAMI.2008.297. (Cited on pages 90 and 101)

Axel Pinz. Object categorization. *Found. Trends. Comput. Graph. Vis.*, 1(4):255–353, 2005. doi:10.1561/0600000003. (Cited on page 115)

John C. Platt. Probabilistic Outputs for Support Vector Machines and Comparisons to Regularized Likelihood Methods. In *Advances in Large Margin Classifiers*, pages 61–74. MIT Press, 1999. (Cited on page 51)

Esther-Sabrina Platzer and Joachim Denzler. Combining Structure and Appearance for Anomaly Detection in Wire Ropes. In *Proceedings of the International Conference on Computer Analysis of Images and Patterns*. 2011. (Cited on pages 168, 169, 170 and 172)

Esther-Sabrina Platzer, Joachim Denzler, Herbert Süße, Josef Nägele, and Karl-Heinz Wehking. Challenging Anomaly Detection in Wire Ropes Using Linear Prediction Combined with One-class Classification. In *Proceedings of the 13th International Workshop on Vision, Modelling, and Visualization (VMV)*, pages 343–352. 2008. (Cited on page 168)

Esther-Sabrina Platzer, Joachim Denzler, Herbert Süße, Josef Nägele, and Karl-Heinz Wehking. Robustness of Different Features for One-class Classification and Anomaly Detection in Wire Ropes. In *Proceedings of the 4th International Conference on Computer Vision, Theory and Applications (VISAPP)*, volume 1, pages 171–178. 2009a. (Cited on page 168)

Esther-Sabrina Platzer, Josef Nägele, Karl-Heinz Wehking, and Joachim Denzler. HMM-based Defect Localization in Wire Ropes - A new Approach to Unusual Subsequence Recognition. In *Proceedings of the 2009 Annual Symposium of the German Association for Pattern Recognition (DAGM'09)*, pages 442–451. 2009b. (Cited on pages 169, 172 and 173)

Esther-Sabrina Platzer, Herbert Süße, Josef Nägele, Karl-Heinz Wehking, and Joachim Denzler. On the Suitability of Different Features for Anomaly Detection in Wire Ropes. In *Computer Vision, Imaging and Computer Graphics: Theory and Applications*, volume 68, pages 296–308. 2010. (Cited on pages 168, 170, 171 and 172)

Jean Ponce, T. L. Berg, M. Everingham, D. Forsyth, M. Hebert, Svetlana Lazebnik, Marcin Marszałek, Cordelia Schmid, C. Russell, A. Torralba, C. Williams, Jianguo Zhang, and Andrew Zisserman. Dataset issues in object recognition. In *Towards Category-Level Object Recognition*, pages 29–48. Springer, 2006. (Cited on pages 98 and 137)

Foster J. Provost, Tom Fawcett, and Ron Kohavi. The Case against Accuracy Estimation for Comparing Induction Algorithms. In *Proceedings of the 1998 International Conference on Machine Learning (ICML'98)*, ICML '98, pages 445–453. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1998. (Cited on page 133)

Ariadna Quattoni, Michael Collins, and Trevor Darrell. Learning Visual Representations using Images with Captions. In *Proceedings of the 2007 IEEE Conference on Computer Vision and Pattern Recognition (CVPR'07)*, pages 1–8. 2007. doi:10.1109/CVPR.2007.383173. (Cited on pages 13 and 81)

Ariadna Quattoni, Michael Collins, and Trevor Darrell. Transfer learning for image classification with sparse prototype representations. In *Proceedings of the 2008 IEEE Conference on Computer Vision and Pattern Recognition (CVPR'08)*, pages 1–8. 2008. doi:10.1109/CVPR.2008.4587637. (Cited on page 13)

Andrew Rabinovich, Andrea Vedaldi, Carolina Galleguillos, Eric Wiewiora, and Serge Belongie. Objects in Context. In *Proceedings of the 2007 International Conference on Computer Vision (ICCV'07)*, pages 1–8. 2007. doi:10.1109/ICCV.2007.4408986. (Cited on page 17)

R. Raina, A. Ng, and D. Koller. Transfer learning by constructing informative priors. In *Proceedings of the 2006 International Conference on Machine Learning (ICML'06)*. 2006. (Cited on page 14)

Carl Edward Rasmussen and Christopher K. I. Williams. *Gaussian Processes for Machine Learning (Adaptive Computation and Machine Learning)*. The

MIT Press, 2005. ISBN 026218253X. (Cited on pages 14, 20, 31, 43, 51, 52, 60, 61, 62, 63, 66, 67, 69, 96, 149, 189 and 190)

Philip Resnik. Using information content to evaluate semantic similarity in a taxonomy. In *Proceedings of the 14th International Joint Conference on Artificial Intelligence*, pages 448–453. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1995. (Cited on page 97)

Erik Rodner and Joachim Denzler. Learning with Few Examples using a Constrained Gaussian Prior on Randomized Trees. In *Proceedings of the Vision, Modelling, and Visualization Workshop*, pages 159–168. Konstanz, October 2008. (Cited on pages 74 and 76)

Erik Rodner and Joachim Denzler. Learning with Few Examples by Transferring Feature Relevance. In *Proceedings of the 2009 Annual Symposium of the German Association for Pattern Recognition (DAGM'09)*, pages 252–261. 2009a. (Cited on page 82)

Erik Rodner and Joachim Denzler. Randomized Probabilistic Latent Semantic Analysis for Scene Recognition. In *Proceedings of the 14th Iberoamerican Congress on Pattern Recognition (CIARP)*, pages 945–953. 2009b. (Cited on pages 22, 23 and 122)

Erik Rodner and Joachim Denzler. One-Shot Learning of Object Categories using Dependent Gaussian Processes. In *Proceedings of the 2010 Annual Symposium of the German Association for Pattern Recognition (DAGM'10)*, pages 232–241. 2010. (Cited on pages 88 and 145)

Erik Rodner and Joachim Denzler. Learning with Few Examples for Binary and Multiclass Classification Using Regularization of Randomized Trees. *Pattern Recognition Letters*, 32(2):244–251, 2011. doi:10.1016/j.patrec.2010.08.009. (Cited on page 74)

Erik Rodner, Doaa Hegazy, and Joachim Denzler. Multiple Kernel Gaussian Process Classification for Generic 3D Object Recognition From Time-of-Flight Images. In *Proceedings of the International Conference on Image and Vision Computing*. 2010. (Cited on pages 69, 120 and 176)

Erik Rodner, Esther-Sabrina Wacker, Michael Kemmler, and Joachim Denzler. One-Class Classification for Anomaly Detection in Wire Ropes with Gaussian

Processes in a Few Lines of Code. In *Proceedings of the 12th IAPR Conference on Machine Vision Applications (MVA)*, pages 219–222. 2011. (Cited on page 167)

Jeremy Rogers and Steve R. Gunn. Identifying Feature Relevance Using a Random Forest. In *Subspace, Latent Structure and Feature Selection, Statistical and Optimization, Perspectives Workshop*, pages 173–184. 2005. (Cited on page 87)

M. Rohrbach, M. Stark, G. Szarvas, and B. Schiele. Combining Language Sources and Robust Semantic Relatedness for Attribute-Based Knowledge Transfer. In *Parts and Attributes Workshop at the European Conference on Computer Vision (ECCV)*. 2010a. (Cited on page 16)

Marcus Rohrbach, Michael Stark, and Bernt Schiele. Evaluating Knowledge Transfer and Zero-Shot Learning in a Large-Scale Setting. In *Proceedings of the 2011 IEEE Conference on Computer Vision and Pattern Recognition (CVPR'11)*. 2011. (Cited on page 16)

Marcus Rohrbach, Michael Stark, Gyö Szarvas, Bernt Schiele, and Iryna Gurevych. What Helps Where - And Why? Semantic Relatedness for Knowledge Transfer. In *Proceedings of the 2010 IEEE Conference on Computer Vision and Pattern Recognition (CVPR'10)*, pages 910–917. 2010b. doi:10.1109/CVPR.2010.5540121. (Cited on pages 16, 97 and 155)

Volker Roth. Kernel Fisher Discriminants for Outlier Detection. *Neural Computation*, 18:942–960, April 2006. doi:10.1162/089976606775774679. (Cited on page 20)

Bryan C. Russell, Antonio Torralba, Kevin P. Murphy, and William T. Freeman. LabelMe: A Database and Web-Based Tool for Image Annotation. *International Journal of Computer Vision*, 77(1-3):157–173, 2008. doi:10.1007/s11263-007-0090-8. (Cited on page 2)

K. Saenko, B. Kulis, M. Fritz, and T. Darrell. Adapting Visual Category Models to New Domains. In *Proceedings of the 2010 European Conference on Computer Vision (ECCV'10)*, pages 213–226. 2010. (Cited on page 12)

Ruslan Salakhutdinov, Antonio Torralba, and Josh Tenenbaum. Learning to Share Visual Appearance for Multiclass Object Detection. In *Proceedings*

of the 2011 IEEE Conference on Computer Vision and Pattern Recognition (CVPR'11). 2011. (Cited on page 13)

S. Savarese, A. Del Pozo, JC. Niebles, and L. Fei-Fei. Spatial-Temporal Correlations for Unsupervised Action Classification. In *IEEE Workshop on Motion and Video Computing*. 2008. doi:10.1109/WMVC.2008.4544068. (Cited on page 165)

Bernhard Schölkopf, John C. Platt, John C. Shawe-Taylor, Alex J. Smola, and Robert C. Williamson. Estimating the Support of a High-Dimensional Distribution. *Neural Computation*, 13(7):1443–1471, 2001. doi:10.1162/089976601750264965. (Cited on pages 20, 113 and 114)

Christian Schuldt, Ivan Laptev, and Barbara Caputo. Recognizing Human Actions: A Local SVM Approach. In *Proceedings of the 2004 International Conference on Pattern Recognition (ICPR'04)*, pages 32–36. 2004. doi:10.1109/ICPR.2004.747. (Cited on pages 163 and 165)

A. Schwaighofer, V. Tresp, and K. Yu. Learning Gaussian process kernels via hierarchical Bayes. In *Advances in Neural Information Processing Systems*, volume 17, pages 1209–1216. Citeseer, 2005. (Cited on pages 15 and 100)

Bernhard Schölkopf and Alexander J. Smola. *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. MIT Press, Cambridge, MA, USA, 2001. ISBN 0262194759. (Cited on pages 6, 38, 39, 41, 44, 45, 46, 47, 49, 50, 193 and 195)

D. W. Scott and S. R. Sain. *Multi-Dimensional Density Estimation*, pages 229–263. Elsevier, Amsterdam, 2004. (Cited on pages 20, 109 and 114)

M. Seeger. *Bayesian Gaussian Process Models: PAC-Bayesian Generalisation Error Bounds and Sparse Approximations*. Ph.D. thesis, University of Edinburgh, 2003. (Cited on pages 52 and 55)

Matthias W. Seeger. Bayesian Inference and Optimal Design for the Sparse Linear Model. *Journal of Machine Learning Research*, 9:759–813, June 2008. (Cited on page 6)

Shai Shalev-Shwartz, Yoram Singer, and Andrew Y. Ng. Online and batch learning of pseudo-metrics. In *Proceedings of the 2004 International Conference on Machine Learning (ICML'04)*, pages 94–101. 2004. doi:10.1145/1015330.1015376. (Cited on page 12)

J. Shawe-Taylor, M. Anthony, and N.L. Biggs. Bounding Sample Size with the Vapnik-Chervonenkis Dimension. *Discrete Applied Mathematics*, 42(1):65–73, 1993. (Cited on page 5)

Jamie Shotton, Matthew Johnson, and Roberto Cipolla. Semantic Texton Forests for Image Categorization and Segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR'08)*, pages 1–8. 2008. (Cited on pages 16, 37 and 124)

Noah Snavely, Steven M. Seitz, and Richard Szeliski. Photo tourism: Exploring photo collections in 3D. In *Proceedings of the SIGGRAPH conference*, pages 835–846. ACM Press, New York, NY, USA, 2006. (Cited on page 116)

Sören Sonnenburg, Gunnar Rätsch, Christin Schäfer, and Bernhard Schölkopf. Large Scale Multiple Kernel Learning. *Journal of Machine Learning Research*, 7:1531–1565, December 2006. (Cited on page 51)

Michael Stark, Michael Goesele, and Bernt Schiele. A Shape-based Object Class Model for Knowledge Transfer. In *Proceedings of the 2009 International Conference on Computer Vision (ICCV'09)*, pages 373–380. 2009. (Cited on page 14)

J. A. K. Suykens, T. Van Gestel, J. De Brabanter, B. De Moor, and J. Vandewalle. *Least Squares Support Vector Machines*. World Scientific Pub. Co., 2002. (Cited on pages 65, 102 and 113)

Richard Szeliski. *Computer Vision - Algorithms and Applications*. Springer, 2011. doi:10.1007/978-1-84882-935-0. (Cited on page 115)

Xiaoyang Tan, Songcan Chen, Zhi-Hua Zhou, and Fuyan Zhang. Face recognition from a single image per person: A survey. *Pattern Recognition*, 39(9):1725–1745, 2006. doi:10.1016/j.patcog.2006.03.013. (Cited on page 3)

K. Tang, M. Tappen, R. Sukthankar, and C. Lampert. Optimizing One-Shot Recognition with Micro-Set Learning. In *Proceedings of the 2010 IEEE Conference on Computer Vision and Pattern Recognition (CVPR'10)*, pages 3027–3034. 2010. doi:10.1109/CVPR.2010.5540053. (Cited on page 81)

L. Tarassenko, P. Hayton, N. Cerneaz, and M. Brady. Novelty detection for the identification of masses in mammograms. In *Fourth International Conference on Artificial Neural Networks*, pages 442–447. 1995. (Cited on page 18)

David M. J. Tax. *One-class classification*. Ph.D. thesis, Delft University of Technology, June 2001. (Cited on pages 19, 21, 104 and 105)

David M. J. Tax and Robert P. W. Duin. Support Vector Data Description. *Machine Learning*, 54(1):45–66, 2004. doi:10.1023/B:MACH.0000008084.60811.49. (Cited on pages 20, 104, 112, 113, 188 and 191)

David M. J. Tax and Piotr Juszczak. Kernel Whitening for One-Class Classification. In *SVM '02: Proceedings of the First International Workshop on Pattern Recognition with Support Vector Machines*, pages 40–52. Springer-Verlag, London, UK, 2002. (Cited on page 111)

D.M.J. Tax and R.P.W. Duin. Uniform object generation for optimizing one-class classifiers. *Journal of Machine Learning Research*, 2:155–173, 2002. (Cited on page 105)

Y. W. Teh, M. Seeger, and M. I. Jordan. Semiparametric Latent Factor Models. In *Proceedings of the International Workshop on Artificial Intelligence and Statistics*, volume 10. 2005. (Cited on pages 15 and 101)

Sebastian Thrun. Is Learning The n-th Thing Any Easier Than Learning The First? In *Advances in Neural Information Processing Systems*, volume 8, pages 640–646. 1996. (Cited on page 12)

Sebastian Thrun and Joseph O'Sullivan. Discovering Structure in Multiple Learning Tasks: The TC Algorithm. In *Proceedings of the 1996 International Conference on Machine Learning (ICML'96)*, pages 489–497. 1996. (Cited on page 17)

Sebastian Thrun and L. Pratt. *Learning To Learn*. Kluwer Academic Publishers, November 1997. (Cited on pages 7 and 11)

Roberto Toldo, Umberto Castellani, and Andrea Fusiello. A Bag of Words Approach for 3D Object Categorization. In *MIRAGE '09: Proceedings of the*

4th International Conference on Computer Vision/Computer Graphics Collaboration Techniques, pages 116–127. Springer-Verlag, Berlin, Heidelberg, 2009. doi:10.1007/978-3-642-01811-4_11. (Cited on page 177)

T. Tommasi, F. Orabona, and B. Caputo. Safety in Numbers: Learning Categories from Few Examples with Multi Model Knowledge Transfer. In *Proceedings of the 2010 IEEE Conference on Computer Vision and Pattern Recognition (CVPR'10)*, pages 3081–3088. 2010. (Cited on pages 17, 103, 135, 144, 145, 146 and 184)

Tatiana Tommasi and Barbara Caputo. The more you know, the less you learn: from knowledge transfer to one-shot learning of object categories. In *Proceedings of the 2009 British Machine Vision Conference (BMVC'09)*. 2009. (Cited on pages 17, 89, 96, 102, 103, 135, 144, 145, 146 and 149)

A. Torralba, R. Fergus, and W. T. Freeman. 80 million tiny images: a large dataset for non-parametric object and scene recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(11):1958–1970, 2008. (Cited on page 2)

A. Torralba and A. Oliva. Statistics of Natural Image Categories. *Network: Computation in Neural Systems*, 14(1):391–412, 2003. (Cited on page 156)

A. Torralba, B.C. Russell, and J. Yuen. LabelMe: online image annotation and applications. *Proceedings of the IEEE*, 98(8):1467–1484, 2010. (Cited on page 2)

Antonio Torralba. Contextual Priming for Object Detection. *International Journal of Computer Vision*, 53(2):169–191, 2003. doi:10.1023/A:1023052124951. (Cited on page 116)

Antonio Torralba and Alexei A. Efros. Unbiased Look at Dataset Bias. In *Proceedings of the 2011 IEEE Conference on Computer Vision and Pattern Recognition (CVPR'11)*. 2011. (Cited on page 137)

Antonio Torralba, Kevin P. Murphy, and William T. Freeman. Sharing Visual Features for Multiclass and Multiview Object Detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(5):854–869, 2007. doi:10.1109/TPAMI.2007.1055. (Cited on pages 13, 75, 88 and 94)

Muhammad Muneeb Ullah, Sobhan Naderi Parizi, and Ivan Laptev. Improving Bag-of-Features Action Recognition with Non-Local Cues. In *Proceedings of the British Machine Vision Conference*, pages 95.1–95.11. BMVA Press, 2010. doi:10.5244/C.24.95. (Cited on page 43)

S. Ullman, M. Vidal-Naquet, and E. Sali. Visual features of intermediate complexity and their use in classification. *Nature Neuroscience*, 5(7):682–687, 2002. (Cited on page 122)

R. Urtasun, A. Quattoni, N. D. Lawrence, and T. Darrell. Transfering Nonlinear Representations using Gaussian Processes with a Shared Latent Space. In *Proceedings of the Learning Workshop (Snowbird)*. Utah, 2008. (Cited on pages 13 and 101)

Raquel Urtasun and Trevor Darrell. Local Probabilistic Regression for Activity-Independent Human Pose Inference. In *Proceedings of the 2008 IEEE Conference on Computer Vision and Pattern Recognition (CVPR'08)*, pages 1–8. 2008. doi:10.1109/CVPR.2008.4587360. (Cited on page 191)

Koen E. A. van de Sande, Theo Gevers, and Cees G.M. Snoek. Evaluating Color Descriptors for Object and Scene Recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(9):1582–1596, 2010. (Cited on pages 116, 117, 119, 120, 127, 147 and 179)

J. van de Weijer, T. Gevers, and J. M. Geusebroek. Edge and Corner Detection by Photometric Quasi-Invariants. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(4):625–630, 2005. (Cited on page 119)

Joost van de Weijer and Cordelia Schmid. Coloring Local Feature Extraction. In *Proceedings of the 2006 European Conference on Computer Vision (ECCV'06)*, pages 334–348. 2006. doi:10.1007/11744047_26. (Cited on page 119)

L.J.P. van der Maaten and G.E. Hinton. Visualizing High-Dimensional Data Using t-SNE. *ournal of Machine Learning Research*, 9:2579–2605, 2008. (Cited on pages 169 and 171)

V.N. Vapnik. *The nature of statistical learning theory*. Springer, 1995. (Cited on pages 19 and 27)

V.N. Vapnik. *The nature of statistical learning theory*. Springer, 2000. (Cited on pages 6, 16 and 41)

A. Vedaldi, G. Guidi, and S. Soatto. Joint Alignment up to (Lossy) Transformations. In *Proceedings of the 2008 IEEE Conference on Computer Vision and Pattern Recognition (CVPR'08)*. 2008. doi:10.1109/CVPR.2008.4587781. (Cited on page 12)

A. Vedaldi and S. Soatto. A Complexity-Distrotion Approach to Joint Pattern Alignment. In *Advances in Neural Information Processing Systems*. 2007. (Cited on page 12)

A. Vedaldi and S. Soatto. Relaxed Matching Kernels for Object Recognition. In *Proceedings of the 2008 IEEE Conference on Computer Vision and Pattern Recognition (CVPR'08)*. 2008. doi:10.1109/CVPR.2008.4587619. (Cited on pages 128 and 160)

Sreekanth Vempati, Andrea Vedaldi, Andrew Zisserman, and C. V. Jawahar. Generalized RBF feature maps for Efficient Detection. In *Proceedings of the British Machine Vision Conference*, pages 2.1–2.11. BMVA Press, 2010. doi:10.5244/C.24.2. (Cited on page 49)

Régis Vert and Jean-Philippe Vert. Consistency and Convergence Rates of One-Class SVMs and Related Algorithms. *Journal for Machine Learning Research*, 7:817–854, 2006. (Cited on page 114)

Michel Vidal-Naquet and Shimon Ullman. Object Recognition with Informative Features and Linear Classification. In *Proceedings of the 2003 International Conference on Computer Vision (ICCV'03)*, pages 281–288. 2003. doi:10.1109/ICCV.2003.1238356. (Cited on page 122)

Paul Viola and Michael Jones. Robust Real-time Object Detection. *International Journal of Computer Vision*, 57(2):137–154, May 2004. doi:10.1023/B:VISI.0000013087.49260.fb. (Cited on page 16)

Esther-Sabrina Wacker and Joachim Denzler. An Analysis-by-Synthesis Approach to Rope Condition Monitoring. In *Proceedings of the 6th International Symposium on Visual Computing (ISVC)*, pages 459–468. 2010. (Cited on pages 169 and 192)

Esther-Sabrina Wacker and Joachim Denzler. Combining Structure and Appearance for Anomaly Detection in Wire Ropes. In *Proceedings of the International Conference on Computer Analysis of Images and Patterns (CAIP'11)*. 2011. (Cited on page 4)

Gang Wang, David Forsyth, and Derek Hoiem. Comparative object similarity for improved recognition with few or no examples. In *Proceedings of the 2010 IEEE Conference on Computer Vision and Pattern Recognition (CVPR'10)*, pages 3525–3532. 2010. doi:10.1109/CVPR.2010.5539955. (Cited on pages 2 and 16)

Gang Wang, Ye Zhang, and Li Fei-Fei. Using Dependent Regions for Object Categorization in a Generative Framework. In *Proceedings of the 2006 IEEE Conference on Computer Vision and Pattern Recognition (CVPR'06)*, pages 1597–1604. 2006. doi:10.1109/CVPR.2006.324. (Cited on page 124)

Jutta Willamowski, Damian Arregui, Gabriella Csurka, Christopher R. Dance, and Lixin Fan. Categorizing Nine Visual Classes using Local Appearance Descriptors. In *ICRPR Workshop on Learning for Adaptable Visual Systems*. 2004. (Cited on page 121)

Shu-Fai Wong and Roberto Cipolla. Extracting Spatiotemporal Interest Points using Global Information. In *Proceedings of the 2007 International Conference on Computer Vision (ICCV'07)*, pages 1–8. 2007. doi:10.1109/ICCV.2007.4408923. (Cited on page 165)

Changchang Wu. SiftGPU: A GPU Implementation of Scale Invariant Feature Transform (SIFT). `http://cs.unc.edu/~ccwu/siftgpu`, 2007. (Cited on page 128)

Ya Xue, Xuejun Liao, Lawrence Carin, and Balaji Krishnapuram. Multi-Task Learning for Classification with Dirichlet Process Priors. *Journal of Machine Learning Research*, 8:35–63, May 2007. (Cited on page 9)

Makoto Yamada, Masashi Sugiyama, and Tomoko Matsui. Semi-supervised speaker identification under covariate shift. *Signal Processing*, 90(8):2353–2361, August 2010. doi:10.1016/j.sigpro.2009.06.001. (Cited on page 3)

Liu Yang and Rong Jin. Distance Metric Learning: A Comprehensive Survey. Technical report, Department of Computer Science and Engineering, Michigan State University, 2006. doi:10.1.1.91.4732. (Cited on page 12)

Kai Yu and Wei Chu. Gaussian Process Models for Link Analysis and Transfer Learning. In J.C. Platt, D. Koller, Y. Singer, and S. Roweis, editors, *Advances in Neural Information Processing Systems*, pages 1657–1664. MIT Press, Cambridge, MA, 2008. (Cited on pages 15 and 101)

Shipeng Yu, Volker Tresp, and Kai Yu. Robust multi-task learning with t-processes. In *Proceedings of the 2007 international conference on Machine learning (ICML'07)*, pages 1103–1110. ACM, New York, NY, USA, 2007. doi:10.1145/1273496.1273635. (Cited on page 101)

D. L. Zhang, Y. N. Cao, C. Wang, and D. G. Xu. A New Method of Defects Identification for Wire Rope Based on Three-Dimensional Magnetic Flux Leakage. *Journal of Physics Conference Series*, 48:334–338, 2006. doi:10.1088/1742-6596/48/1/062. (Cited on pages 167 and 169)

Guangpeng Zhang and Yunhong Wang. Faceprint: Fusion of Local Features for 3D Face Recognition. In *ICB '09: Proceedings of the Third International Conference on Advances in Biometrics*, pages 394–403. Springer-Verlag, Berlin, Heidelberg, 2009. doi:10.1007/978-3-642-01793-3_41. (Cited on page 177)

J. Zhang, M. Marszalek, S. Lazebnik, and C. Schmid. Local Features and Kernels for Classification of Texture and Object Categories: A Comprehensive Study. *International Journal of Computer Vision*, 73(2):213–238, 2007. doi:10.1007/s11263-006-9794-4. (Cited on pages 119 and 176)

Y. Zhang and D.Y. Yeung. Semi-Supervised Multi-Task Regression. In *Proceedings of the European Conference on Machine Learning and Knowledge Discovery in Databases: Part II*, pages 617–631. Springer, 2009. doi:10.1007/978-3-642-04174-7_40. (Cited on page 100)

Ziming Zhang, Yiqun Hu, Syin Chan, and Liang-Tien Chia. Motion Context: A New Representation for Human Action Recognition. In *Proceedings of the 2008 European Conference on Computer Vision (ECCV'08)*, pages 817–829. 2008. (Cited on page 163)

Bin Zhao, Li Fei-Fei, and Eric P. Xing. Online Detection of Unusual Events in Videos via Dynamic Sparse Coding. In *Proceedings of the 2011 IEEE Conference on Computer Vision and Pattern Recognition (CVPR'11)*. 2011. (Cited on page 163)

George K. Zipf. *Human Behavior and the Principle of Least Effort*. Addison-Wesley, 1949. (Cited on page 2)

Alon Zweig and Daphna Weinshall. Exploiting Object Hierarchy: Combining Models from Different Category Levels. In *Proceedings of the 2007 International Conference on Computer Vision (ICCV'07)*, pages 1–8. 2007. (Cited on page 74)

Notation

The following table contains the main symbols and notational conventions used in this thesis.

$\overset{*}{=}$	equality that requires certain further assumptions explained in the paragraph next to the equations
$\overset{\text{def}}{=}$	equation related to a definition
n	number of training examples
D	dimension of a feature vector, number of features
$\boldsymbol{x} \in \mathbb{R}^D, x_i \in \mathbb{R}$	vectors are denoted as bold lowercase letters and a single element of a vector is written in normal font
$\mathbf{A} \in \mathbb{R}^{n \times D}, A_{ij} \in \mathbb{R}$	matrices are written as bold uppercase letters
$\text{diag}(\mathbf{A})$	vector containing the diagonal elements of the quadratic matrix $\mathbf{A}$
$\mathbf{I}$	identity matrix, the size depends on the context and is not given explicitly
$\mathcal{O}(f(n))$	Landau notation for specifying asymptotic growth rates
$\delta[x]$	Dirac delta impulse with $\delta[x] = \infty$ for $x = 0$ and zero otherwise
$\delta(x)$	discrete impulse with $\delta(x) = 1$ for $x = 0$ and zero otherwise
$\text{sign}(x)$	returns the sign of the argument, *i.e.* if x is negative it returns -1 and 1 otherwise.
$\{\nabla_{\boldsymbol{x}} f\}(\boldsymbol{x}, \boldsymbol{y})$	gradient vector of the function f with respect to the inputs $\boldsymbol{x}$ (column vector)

$y \in \mathcal{Y}$	single label or output
$\boldsymbol{x} \in \mathcal{X}$	single input or training example
$\boldsymbol{x}_* \in \mathcal{X}$	test input example
$y_* \in \mathcal{Y}$	random variable of the label of the test example $\boldsymbol{x}_*$
$\mathcal{D} = \{(\boldsymbol{x}_1, y_1), \ldots\}$	training set (ordinary set but with a canonical order)
$\mathcal{D}^{\text{test}}, \lvert\mathcal{D}^{\text{test}}\rvert = n^{\text{t}}$	test set used to evaluate the recognition performance of a classification system
M	number of classes or categories
$\boldsymbol{\theta}$	classification model or parameter estimated using training data, for transfer learning algorithms it refers to the information transferred between tasks
$\boldsymbol{w}$	vector of a hyperplane used for a binary SVM classifier
$h : \mathcal{X} \to \mathcal{Y}$	decision function of a classifier, hypothesis
$f : \mathcal{X} \to \mathbb{R}$	soft decision function of a classifier, latent function of a Gaussian process classifier
$\mathcal{H}$	feature space corresponding to the kernel K and the transformation ϕ
$\phi : \mathcal{X} \to \mathcal{H}$	feature transformation
$K : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$	kernel function corresponding to the transformation ϕ (*cf.* Eq. (2.35))
$\mathbf{K} \in \mathbb{R}^{n \times n}$	kernel matrix of the training data $\mathcal{D}$ (*cf.* Definition 2.1)
$\boldsymbol{k}_* \in \mathbb{R}^n$	kernel values computed between the test examples $\boldsymbol{x}_*$ and each of the training examples
$\boldsymbol{\eta}$	hyperparameters of a kernel function
γ	parameter of the Gaussian kernel (Eq. (2.41))
$\boldsymbol{\alpha} \in \mathbb{R}^n$	variable of the dual SVM optimization problem, coefficients of the kernel version of the decision function
r_{B}	fraction of training data used for each base model in a Bagging approach
T	size of an ensemble
$\mathcal{R}$	set of relevant features
m_ℓ	number of leaves in a decision tree

$\vartheta(\boldsymbol{x})$	leaf node in a decision tree corresponding to an input example
H	hinge loss defined as $H(z) = \max(0, 1 - z)$
μ_*	predictive mean of GP regression
σ_*	predictive standard deviation of GP regression
σ_ε	standard deviation of the GP regression noise model
Φ	cumulative Gaussian function used in the probit noise model
σ_c^2	scaling factor of the GP classification probit noise model
τ	target task in a transfer learning setting
s	single support task in a transfer learning setting
$\mathcal{D}^\tau$	training set of the target task/class
$\mathcal{D}^\mathcal{S}$	training set of a single selected support task
$\mathfrak{D}^\mathcal{S}$	training data of all available support tasks
$\boldsymbol{y}_\tau, \boldsymbol{y}_s$	labels of the target and the support task
$\boldsymbol{k}_{\tau *}, \boldsymbol{k}_{s*}$	kernel values computed between a new test example $\boldsymbol{x}_*$ and the set of target and support training examples
ρ	task correlation parameter
$\tilde{\mu}_i, \tilde{\sigma}_i^2$	leave-one-out predictive mean and variance of example $\boldsymbol{x}_i$
$\nu : \mathcal{X} \to \mathbb{R}$	one-class classification score
$\mathcal{L}$	set of local features including descriptors and positions
S	dimension of local descriptors
$\boldsymbol{l} \in \mathbb{R}^S$	multi-dimensional descriptor of a local feature
$\boldsymbol{p}$	position of a local feature
$\boldsymbol{h} \in \mathbb{R}^{n_q}$	bag of visual words histogram
K^{PMK}	pyramid matching kernel
ℓ	levels of the pyramid matching kernel
$\mathbf{C}$	confusion matrix
err-ov	overall recognition rate
err-avg	average recognition rate

List of Figures

List of Tables

List of Theorems and Definitions

Index