Efficient Gaussian Process Classification Using Random Decision Forests[1]

B. Fröhlich, E. Rodner, M. Kemmler, and J. Denzler

Chair for Computer Vision, Friedrich-Schiller University of Jena

e-mail: {bjoern.froehlich, erik.rodner, michael.kemmler, joachim.denzler}@uni-jena.de

Abstract—Gaussian Processes are powerful tools in machine learning which offer wide applicability in regression and classification problems due to their non-parametric and non-linear behavior. However, one of their main drawbacks is the training time complexity which scales cubically with the number of samples. Our work addresses this issue by combining Gaussian Processes with Randomized Decision Forests to enable fast learning. An important advantage of our method is its simplicity and the ability to directly control the trade-off between classification performance and computation speed. Experiments on an indoor place recognition task show that our method can handle large training sets in reasonable time while retaining a good classification accuracy.

DOI: 10.1134/S1054661811020337

INTRODUCTION

Gaussian process (GP) based machine learning techniques have recently gained much attention in the field of pattern recognition and computer vision, since they provide an elegant Bayesian framework and offer state-of-the-art recognition performance [1]. However, one non-negligible disadvantage of the full GP framework is the time (and memory) requirement during learning which scales cubically (and quadratically) with the size of the training data. This fact hinders their use for large scale classification tasks which often occur in object and scene recognition scenarios [2].

In the last decade, Randomized Decision Forests (RDF) [3], which are an extension of Decision Trees, became an increasingly important tool for large scale classification problems. RDFs are basically constructed from classifiers which linearly separate the input space. This simplicity allows for fast learning and testing times but also often accompanies a loss of classification accuracy compared to state-of-the-art classification methods such as Support Vector Machines (SVMs) or Gaussian process classifiers (GPCs).

In this paper we present a learning scheme which combines the benefits from GPCs and RDFs. Previous work trying to speed up GP classification use unsupervised *kd*-trees [4] neglecting label information during clustering. One alternative approach is to enforce some conditional independence structure between latent variables [5]. Other interesting ways for speeding up GP inference can be found in Rasmussen et al. [6]. Our idea is related to [7] which combines GP and Bayesian decision tree models. In contrast we use Random Decision Forests which offer an easy tunable tradeoff between speed and performance.

The remainder of the paper is organized as follows. We briefly review classification and regression with Gaussian processes, which is followed by describing Randomized Decision Forests. The subsequent sections detail our approach of combining these two classifiers and experimentally demonstrate the benefits of this method on a place recognition task [8]. A summary of our findings and a discussion of future research directions conclude the paper.

CLASSIFICATION WITH GAUSSIAN PROCESS PRIORS

In the following we will briefly review Gaussian process regression and classification. Due to the lack of space, we concentrate on the main model assumptions and the resulting prediction equation. For a presentation of the full Bayesian treatment we refer to Rasmussen and Williams [6].

Given n training examples $\mathbf{x}_i \in \mathcal{X}$, which denote input vectors (e.g., images), and corresponding binary labels $y_i \in \{-1, 1\}$, we would like to predict the label y_* of an unseen example $\mathbf{x}_*$. The goal in classification is to find the intrinsic relationship between inputs $\mathbf{x}$ and labels y. It is often assumed that the desired mapping can be modeled by $y = f(\mathbf{x}) + \epsilon$, where f is a noise-free latent function (which is not observed during training) and ϵ denotes a noise term. One common modelling approach is to assume that f belongs to some parametric family and to learn the parameters which best describe the training data. However, the main benefit of the GP framework is the ability to model the underlying function f directly, i.e. without any fixed parameterization (since all parameter configurations are taken into account).

[1]The article is published in the original.

Received December 20, 2010

The two main modelling assumptions for Gaussian process classifiers are as follows:

1. The latent function f is a sample from a Gaussian process (GP) prior

$$f \sim \mathcal{GP}(\mathbf{0}, \mathcal{K}(\cdot, \cdot))$$

with zero mean and covariance or kernel function $\mathcal{K}$.

2. Labels y are conditionally independent given latent function values $f(\mathbf{x})$ and are described using some noise model $p(y|f(\mathbf{x}))$.

The Gaussian process prior enables to model the correlation between labels using the similarity of inputs, which is described by the kernel function. It is thus possible to model the common assumption of smoothness, i.e. that similar inputs should lead to similar labels. For classification purposes often sigmoid functions are employed as noise models [6]. In contrast, we will follow Kapoor et al. [1] and use zero-mean Gaussian noise with variance σ_n^2:

$$p(y|f(\mathbf{x})) = \mathcal{N}(y|f(\mathbf{x}), \sigma_n^2), \qquad (1)$$

which is the standard assumption for GP regression. The advantage of this label regression approach is that tractable predictions for unseen points $\mathbf{x}_*$ are possible, without using approximative inference methods [6].

Let $\mathbf{K}$ be the kernel matrix with pairwise kernel values of the training examples $\mathbf{K}_{ij} = \mathcal{K}(\mathbf{x}_i, \mathbf{x}_j)$ and $\mathbf{k}_*$ be kernel values $(\mathbf{k}_*)_i = \mathcal{K}(\mathbf{x}_i, \mathbf{x}_*)$ corresponding to test example $\mathbf{x}_*$. The most likely outcome $\bar{y}_*$ given input $\mathbf{x}_*$ and labeled training data can then be predicted analytically using the following equation:

$$\bar{y}_*(\mathbf{x}_*) = \mathbf{k}_*^T(\mathbf{K} + \sigma_n^2\mathbf{I})^{-1}\mathbf{y}. \qquad (2)$$

Applying the one-vs.-all strategy in combination with a majority voting scheme, multi-class classification problems can be solved without much additional effort [1].

RANDOMIZED DECISION FOREST

In contrast to GP techniques, a *Randomized Decision Forest* (RDF [3]) is an ensemble classifier that can handle large sets of training examples of high dimensionality. This advantage is mainly due to the simplicity of the linear base classifiers (decision stumps) which cluster the feature space. Compared to standard decision tree approaches, which suffer from severe over-fitting problems, a RDF is an ensemble (forest) of T decision trees generated using randomized learning techniques [9]. Each tree is learned with only a random fraction of the available training data and the data is recursively split by axis orthogonal hyperplanes which are learned by maximizing the information gain of a randomly selected feature set. The procedure is stopped if the current set contains only examples of a single class or the number of examples falls below a certain value l.

Computational Complexity for Balanced Datasets

	training	testing
GP	$\mathcal{O}(n^3)$	$\mathcal{O}(n)$
RDF	$\mathcal{O}(Tn\log n)$	$\mathcal{O}(T\log n)$
RDF-GP	$\mathcal{O}(Tn\log n + Tnl^3)$	$\mathcal{O}(T\log n + Tl)$

To classify a new example $\mathbf{x}_*$, each tree t in the ensemble is traversed until a leaf node $v_t(\mathbf{x}_*)$ is reached. Each such leaf node contains a histogram which describes the posterior probability $p(y_* = \kappa|v_t(\mathbf{x}_*))$ obtained during learning. The final posterior probability is then estimated by simple averaging over all leaf probabilities:

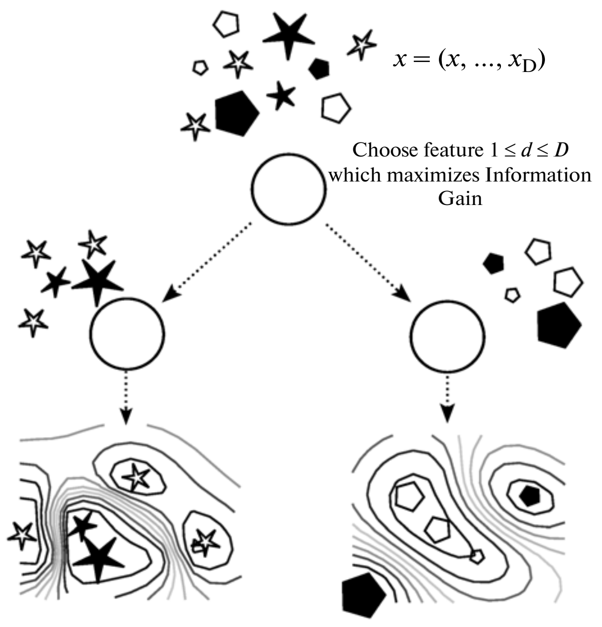
$$p(y_* = \kappa|\mathbf{x}_*) = \frac{1}{T}\sum_{t=1}^{T} p(y = \kappa|v_t(\mathbf{x}_*)). \qquad (3)$$

COMBINING RDF AND GP

For speeding up vanilla GP classifiers, we propose to efficiently combine RDF and GP. In principle, GP classifiers can be utilized as weak classifiers in each node of the underlying decision trees. This strategy, however, would even amplify the computational cost of the learning algorithm. The main idea of our approach is thus to learn a RDF whose inner nodes are simple decision stumps—only leaf nodes are equipped with powerful GP classifiers. This allows to describe the posterior distribution in each leaf using a non-linear decision function and to achieve a significant speed-up due to the clustering of the training data performed by the RDF. From a different perspective, we are constructing an ensemble of GP classifiers where each single classifier is trained on a pre-clustered subset of examples (and features). These subsets are given by $\mathcal{X}_v = \{\mathbf{x}_i \in \mathcal{X}|v_t(\mathbf{x}_i) = v\}$, i.e. the training data reaching the corresponding leaf nodes v. Upper bounds of this method for training and learning are listed in the table, where n is the number of training samples, T is the number of trees and l is the maximum number of examples in a leaf.

EXPERIMENTS

Our method is assessed using the place recognition database utilized in [8] which comprises seven different room categories. The authors of [8] originally utilized 2 sequences (697 samples), captured by a mobile robot, for training and 6 sequences (2759 samples) for testing. Since our work focuses on learning with large quantities, we swap both sets and thus use the larger



$x = (x, ..., x_D)$
Choose feature $1 \leq d \leq D$ which maximizes Information Gain

Fig. 1. The outline of our approach: RDF is used to cluster the data in a supervised manner and a GP classifier is used to separate classes in each leaf node.

fraction for learning the respective classifier models. In our experiments we empirically support the following *hypotheses*:

1. RDF-GP clearly outperforms the standard RDF which only uses decision stumps

2. RDF-GP is substantially faster than the full GP classifier approach

In our experimental setting, we followed [8] and employed a combination of different image-based features using both a standard camera and a time-of-flight sensor. For both sensors, the following features are computed: (1) a bank of reduced Gabor filter responses with 8 different orientations and 4 different scales and (2) 16 slopes and offsets of oriented fourier power spectra. For range images, additional information in terms of (3) range histograms and (4) surface normal histograms are taken into account. All features are concatenated and thus treated as one large feature vector. As covariance functions, we used the common RBF-kernel:

$$\mathcal{K}(\mathbf{x}, \mathbf{x}') = \exp(-\gamma\|\mathbf{x}-\mathbf{x}'\|^2) \quad (4)$$

where γ denotes the bandwidth parameter of the kernel.

For comparison of RDF, GP classifier and our combined approach from the previous section (RDF-GP), average recognition rate is used as an unbiased accuracy measure. In our setting, we utilized RDFs with 5 decision trees and a varying number $l \in \{5, 6, ..., 200\}$ of maximum examples in leaf nodes. The hyper-

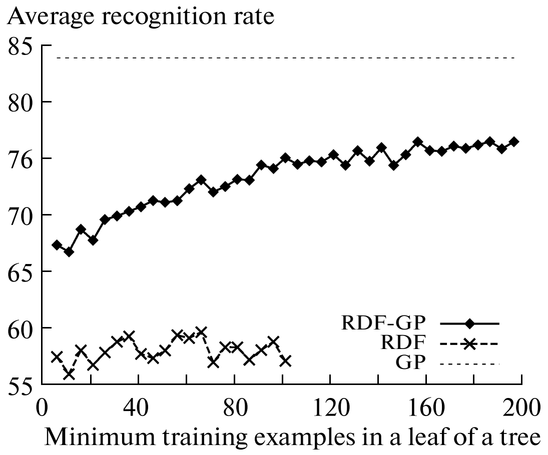


Fig. 2. Average recognition rate for GP classifier, Randomized Decision Forests and RDF-GP (measured in percent).

parameter of the RBF-kernel is set to $\gamma = \exp(5.0)$ in all experiments.

The results of all methods for the place recognition scenario are illustrated in Fig. 3. It can be seen that the GP classifier performs best and RDF clearly exhibits inferior performance compared to the other methods (Recognition rates for l = 100 are not displayed for RDF, since no improvement is made with an increasing number of examples in leaf nodes). By augmenting the RDF with GP classifiers in each node, however, progress in terms of accuracy is achieved. It can be nicely seen that the accuracy increases with increasing l which is due to the fact that more and more training examples are fed into the GP classifiers located in the leaf nodes.

In order to compare the complexity of the respective methods, the runtime behavior for training and

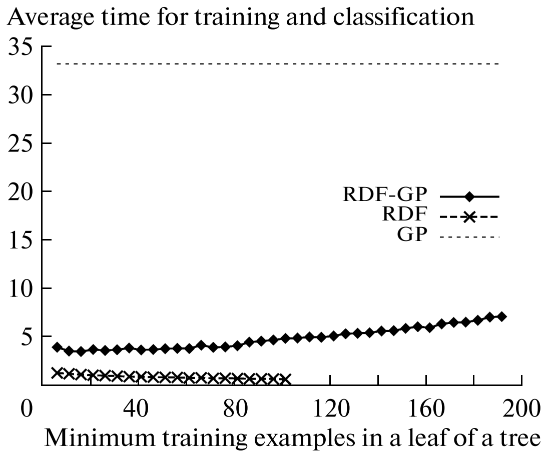


Fig. 3. Average computation time for GP classifier, Random Forests and RDF-GP (measured in seconds).

testing is investigated. The results on a standard office computer (2.80 GHz) are depicted in Fig. 3, where training and testing times are summed up to yield one final value. It becomes clear that, compared to the full GP classifier, RDF and RDF-GP have a substantial runtime advantage. This behavior nicely illustrates the complexity bounds from the table, where RDF and RDF-GP have similar complexity for a moderate number of examples in the leaf nodes.

CONCLUSIONS AND FURTHER WORK

This work presents a new approach for boosting the runtime of Gaussian Process (GP) classifiers, where Randomized Decision Forests are used to compute a pre-clustering of the input space. We empirically validate the combined method on a place recognition task. While retaining a moderate amount of accuracy, our method shows to be substantially faster than the full GP classifier.

In the future, we are planning to do an in-depth runtime analysis for learning with large-scale databases like Imagenet [2]. Further research has to be done by analyzing the memory requirements of our method. Since the size of the covariance matrix scales quadratically with the number of training examples, but only a few correlations needs to be taken into account, a benefit from our method is expected. The principle of our method is very general and allows a variety of methods to be used as leaf classifiers. It would be hence interesting to analyze other types of learners and to investigate the combination of our method with existing sparse GP approaches.

REFERENCES

1. A. Kapoor, K. Grauman, R. Urtasun, and T. Darrell, "Gaussian Processes for Object Categorization," Int. J. Comp. Vision **88**, No. 2, 169–188 (2010).
2. J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-fei, "Imagenet: A Large–Scale Hierarchical Database," in *Proc. Conf. on Computer Vision and Pattern Recognition* (Fontainebleau Miami Beach, 2009).
3. L. Breiman, "Random Forests," Mach. Learn. **45**, No. 1, 5–32 (2001).
4. Y. Shen, A. Ng, and M. Seeger, "Fast Gaussian Process Regression Using kd–Trees," Adv. Neural Inf. Processes Syst. **18**, 1225–1232 (2006).
5. J. Q. Candela and C. E. Rasmussen, "A Unifying View of Space Approximate Gaussian Processes Regression," J. Mach. Learn. Res. **6**, 1939–1959 (2005).
6. C. E. Rasmussen and C. K. I. Williams, *Gaussian Processes for Machine Learning (Adaptive Computation and Machine Learning* (MIT Press, 2005).
7. T. Broderick and R. B. Gramacy, "Treed Gaussian Process Models for Classification," in *Classification as a Tool for Research* (2010), pp. 101–108.
8. M. Kemmler, E. Rodner, and J. Denzler, "Global Context Extraction for Object Recognition Using a Combination of Range and Visual Features," in *Proc. Dyn3D'09: Proc. DAGM 2009 Workshop on Dynamic 3D Imaging* (Jena, 2009), pp. 96–109.
9. P. Geurts, D. Ernst, and L. Wehenkel, "Extremely Randomized Trees," Mach. Learn. **63**, No. 1, 3–42 (2006).