Probabilistic Nodes for Modelling Classification Uncertainty for Random Forest

Florian Baumann, Karsten Vogt, Arne Ehlers, Bodo Rosenhahn
Institut für Informationsverarbeitung
Appelstraße 9a, 30167 Hannover
lastname@tnt.uni-hannover.de

Abstract

In this paper, we propose to enhance the original Random Forest algorithm by introducing probabilistic nodes. Platt Scaling is used to interpret the decision of each node as a probability and was initially developed for calibrating Support Vector Machines. Nowadays it is used to calibrate the output probabilities of decision trees, boosted trees or Random Forest classifiers. In comparison to these approaches, we integrate the Platt Scaling calibration method into the decision process of every node within the ensemble of decision trees. Regarding the original Random Forest, the nodes serve as a guide to predict the path through the tree until reaching a leaf node. In this paper, we interpret the decision as a probability and incorporate more information into the decision process. The proposed approach is evaluated using two well-known machine learning datasets as well as object recognition datasets.

1 Introduction

Random Forest was developed by Leo Breiman [1] and combines the idea of Bagging [2] with a random feature selection proposed by Ho [3, 4] and Amit [5]. A Random Forest consists of an ensemble of binary decision trees while the final result is gathered by using a majority voting that takes the decision of all trees into account.
If we consider a tree in the forest, there are several leaf nodes while each of them holds an estimation about the final decision. Starting at the first node, an unique combination of decisions lead to the leaf and thus to the final decision of the tree. In other words, the voting of a tree depends on the path that an example takes to reach the corresponding leaf node. This path is characterized by several binary decisions without taking other estimates like the reliability, the uncertainty or the confidence into account. More generally, each binary decision serves as a simple signpost that predicts the path through the tree (for instance take the $left$ or $right$ path). See Figure 1 for a typical decision tree. Our observations show that small fluctuations, noise or simply a large intra-class variation might disrupt the correct path leading to a wrong decision. Thus, it is convenient to incorporate more information into the decision of a node leading to a more robust estimation.
Contribution: In this paper, we use a calibration method to interpret the decision of each node as a probability. The decision no longer rests upon a discrete value (in the case of a CART-like decision tree: $left$ or $right$) but on a continuous probability that better represents the reliability and accurateness. Decisions that are very near to the node's threshold get a higher probability than decisions that are far away. In our approach, the final vote of the tree is not based on several binary decisions that might be disturbed but rather by multiplying the probabilities of all nodes on the path. Our proposed method is evaluated on two well-known datasets for object recognition (*GTSRB*, *MNIST*) and on two typical machine learning datasets (*USPS*, *Letter*).

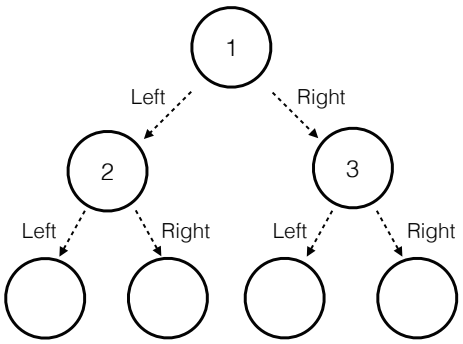

Figure 1. The final decision depends on the path and an unique combination of decisions through the tree. Each node serves as a guide that only tells to take the $left$ or $right$ path.

1.1 Related Work

In [6], Zadrozny and Elkan propose a method to obtain calibrated probability estimates of decision trees and naive Bayesian classifiers by using Isotonic Regression. It turned out that the proposed method is less effective for Random Forest [1].
In [7], Boström observes that even very accurate classifiers like a Random Forest lead to an output probability of a poor quality and proposes a novel calibration method. The author compares the novel calibration method to Platt Scaling and Isotonic Regression and generally shows that a calibrated Random Forest clearly outperforms uncalibrated variants.
In [8], Niculescu-Mizil and Caruana apply Platt Scaling and Isotonic Regression to boosted stumps and boosted decision trees. The authors observe that the probabilities are significantly improved.
In comparison to these works we do not calibrate the output probabilities of a Random Forest directly. Instead, we use the Platt Scaling method to calibrate nodes for obtaining discriminative probabilities that take the uncertainty of the decision into account. Platt Scaling is completely integrated into the decision process of each node. For gathering the final result,

all probabilities on the path to the leaf node are multiplied with the relative class frequency.

2 Random Forest

A Random Forest consists of CART-like decision trees that are independently constructed on a bootstrap sample with randomly chosen features. Random Forest is an ensemble learning method for classification and regression and compared to other ensemble learning algorithms, i.e. Boosting methods [9] that build a flat tree structure of decision stumps, a Random Forest uses an ensemble of unpruned decision trees, is multiclass capable and has some preferable characteristics [1]:

- Similar or better accuracy than AdaBoost.
- Robust to noise and outliers.
- Faster training than bagging or boosting.
- Useful internal estimates: error, strength, correlation and variable importance.

Training procedure: Given a dataset containing N examples for training, $(X,Y) = \{(\vec{x}_1,y_1),...,(\vec{x}_N,y_N)\}$, where $\vec{x}_i$ is the feature vector of M dimensions and y_i is the class label which value is between 1 and K. To grow a tree, the following steps are necessary:

1. Choose n_{tree} examples from the whole training set (X,Y) at random.
2. The remaining examples are used to calculate the out-of-bag error (OOB-error).
3. At each node randomly specify $m_{try} << M$ variables and find the best split.
4. Completely grow the tree to the largest possible extension without pruning.

Classification: A completed Random Forest consists of several classification trees $(1 \leq t \leq T)$ in which the class probabilities, estimated by majority voting, are used to calculate the example's label $y(\vec{x})$ with respect to a feature vector $\vec{x}$:

$$y(\vec{x}) = \underset{c}{\operatorname{argmax}} \left(\frac{1}{T} \sum_{t=1}^{T} I_{h_t(\vec{x})=c} \right) \tag{1}$$

The decision function $h_t(\vec{x})$ provides the classification of a tree to a class c with the indicator function I:

$$I_{h_t(\vec{x})=c} = \begin{cases} 1, & h_t(\vec{x}) = c, \\ 0, & \text{otherwise.} \end{cases} \tag{2}$$

An example is classified by passing it down each tree until a leaf node is reached. A classification result is assigned to each leaf node and the final decision is determined by taking the class having the most votes, see Equation (1).

Dataset	# Train	# Test	# Classes
GTSRB	39209	12630	43
MNIST	60000	10000	10
USPS	7291	2007	10
Letter	16000	4000	26

Table 1. Properties of the object recognition and machine learning datasets.

3 Platt Scaling

Platt Scaling was developed by John Platt in 1999 [10] and originally developed for calibrating Support Vector Machines. The main idea is to transform the outputs of a classification model into a probability distribution. Platt Scaling works by fitting a sigmoid function into the feature space:

$$P(y|x) = \frac{1}{1 + e^{\alpha x + \beta}}, \tag{3}$$

where $P(y|x)$ is the probability that an example with a value of x belongs to class y. α and β are scalar parameters of the sigmoid function that are estimated using a maximum likelihood method that optimizes the training set. The method works by a gradient descent search that minimizes a particular loss function (see [10] for more details).
In this work we use the Platt Scaling variant proposed by Lin et al. [11].

4 Proposed Method

While constructing the tree, the scalar parameters α and β of the sigmoid function are estimated using a gradient descent algorithm. We propose to include the Platt Scaling method directly into the decision process of each node and map a sigmoid function into the corresponding feature space of the node. The original training procedure as described in Section 2 is enhanced to:

Training procedure:

1. Choose n_{tree} examples from the whole training set (X,Y) at random.
2. The remaining examples are used to calculate the out-of-bag error (OOB-error).

Method	Error rate (%)	#Trees
Proposed method	$\mathbf{2.55 \pm 0.10}$	100
Schulter et al. [12]	2.71 ± 0.10	100
RF 4.6-7 in R	3.0	50
Bernard et al. [13]	4.61	500
Fan et al. [14]	4.95	200
Bernard et al. [15]	5.92	210
Bernard et al. [16]	6.73	300

Table 2. Our proposed framework in comparison to *Random Forest based approaches* on the MNIST dataset. The error rate with standard deviation, training time and the number of trees is illustrated.

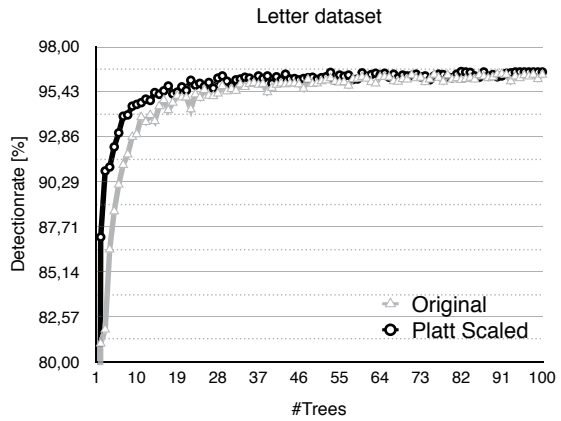

Figure 2. The number of trees with respect to the detection rate using the Letter dataset. Our proposed framework clearly outperforms the original Random Forest algorithm, especially at a smaller number of trees.

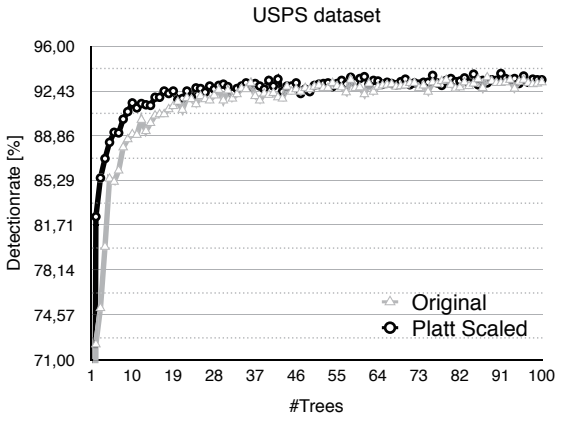

Figure 3. The number of trees with respect to the detection rate using the USPS dataset. Our proposed framework clearly outperforms the original Random Forest algorith, especially at a smaller number of trees.

3. At each node randomly specify $m_{try} << M$ variables and find the best split.
4. Re-label training data according to the original split: $Left_{path} = 1$ and $Right_{path} = -1$.
5. Use Platt Scaling algorithm by [11] to estimate α and β. For each node, only these parameters are saved.
6. Completely grow the tree to the largest possible extension without pruning.

We mention that the proposed method does not lead to an increasing computation time while classifying. The sigmoid parameters α and β are estimated in the offline learning procedure and saved for classifying new examples.

Classification: A new example arrives at the first node and the probability distribution is initially set to the relative class frequency of the corresponding leaf node. For all nodes N on the path of reaching the leaf node, the probability $P(y|x)_{n\cdots N}$ is computed using Equation (3) with the corresponding feature value x and the a priori computed scalar parameters α and β. The final probability distribution for a leaf node is computed by multiplying the relative class frequency with $P(y|x)_{n\cdots N}$.

5 Experimental Results

Our proposed method is evaluated using well-known datasets for machine learning and object recognition. We use the German Traffic Sign Recognition Benchmark dataset GTSRB) [17] and the MNIST dataset for handwritten digit recognition [18]. Further, two standard machine learning datasets (USPS and Letter) are chosen [19, 20]. Table 1 provides an overview about the properties.

5.1 Experimental Setup

The **GTSRB** and **MNIST** dataset consist of a fixed number of training- and test images. All experiments were repeated five times. For both experiments, the raw pixel values without preprocessing are directly used as features for constructing the tree. Better results might be achieved by using more suitable features like HOG features. The **USPS** and **Letter** dataset consist of a fixed number of training- and test examples too. The experiments were repeated five times. For a fair comparison, the number of trees is set to 100 and the number of random splits per node to $\sqrt{NumberOfVariables}$.

5.2 Comparison to State-of-the-art

GTSRB: The Platt Scaled Random Forest algorithm achieves an accuracy of about 89.00% while the original Random Forest algorithm achieves an accuracy of about 88.00%. If we compare our method to the related work by regarding the competition results table[1], other researchers report results at 99.98% by using more complex features and other machine learning algorithms. Thus, a fair comparison is not possible.
MNIST: Table 2 presents the results using the MNIST dataset for handwritten digit recognition. Our proposed method achieves the lowest rate of 2.55% in comparison to related Random Forest based approaches and especially to a well-established implementation in *R*.
USPS and **Letter**: Figures 2 and 3 compare our Platt Scaled Random Forest to the original Random Forest algorithm with respect to the detection rate and number of trees. Our proposed methods outperforms the standard Random Forest algorithm. Especially for a smaller number of trees, the improvement is much larger (up to 6%). This behavior is obvious because of the reliability which is introduced into the decision process of each node: The original Random Forest algorithm requires a certain number of trees for a meaningful and discriminative decision while the Platt Scaled Random Forest additionally uses the confidence of each node leading earlier to a better decision. Table 3 illustrates the results on the USPS and Letter dataset in comparison to the approach of Schulter et al, Boosted

[1] http://benchmark.ini.rub.de

Method	USPS	Letter
Random Forest [12]	5.96 ± 0.21	4.75 ± 0.10
Boosted Trees [21]	5.93 ± 0.27	4.70 ± 0.18
Schulter et al. [12]	5.59 ± 0.16	3.52 ± 0.12
Proposed method	$\mathbf{5.49 \pm 0.15}$	$\mathbf{3.45 \pm 0.13}$

Table 3. Our proposed framework in comparison to the Alternating Random Forest approach of Schulter et al., Boosted Trees and to a standard Random Forest implementation.

Trees [21] and to a standard Random Forest implementation. Our proposed method achieves the lowest error rate of 5.49% using the USPS dataset and an error rate of 3.45% on the Letter dataset.

6 Conclusions

In this paper, we propose probabilistic nodes for the Random Forest classifier. Originally, a node of a decision tree acts like a guide that predicts the path through the tree. This node only decides either to take the *left* or *right* path to reach the leaf node. Our idea is to make this decision more reliable and accurate by incorporating more information like the uncertainty into the decision process. For this task, Platt Scaling is well suited because it fits a sigmoid function to a training set and gives the opportunity to interpret feature values as probabilities. For all nodes on the path, the probabilities are multiplied with the relative class frequency of the corresponding leaf node. Thus, the final probability distribution gives a better estimation than simply taking the class with the most number of examples.

Our proposed approach is evaluated on the GTSRB dataset for traffic sign recognition, on the MNIST dataset for handwritten digit recognition and using the USPS and Letter machine learning datasets. Our method achieves competing results in comparison to related Random Forest based approaches and to state-of-the-art methods. Especially for a smaller number of trees, the probabilistic nodes clearly outperform the original Random Forest algorithm by reaching an improvement of up to 6%. This contribution is significant for resource-limited platforms like mobile devices or driver assistance systems.

References

[1] L. Breiman, "Random forests," in *Machine Learning*, vol. 45, pp. 5–32, 2001.

[2] L. Breiman, "Bagging predictors," in *Machine Learning*, vol. 24, pp. 123–140, 1996.

[3] T. K. Ho, "Random decision forests," in *Document Analysis and Recognition, 1995., Proceedings of the Third International Conference on*, vol. 1, pp. 278–282, IEEE, 1995.

[4] T. K. Ho, "The random subspace method for constructing decision forests," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 20, no. 8, pp. 832–844, 1998.

[5] Y. Amit and D. Geman, "Shape quantization and recognition with randomized trees," *Neural computation*, vol. 9, no. 7, pp. 1545–1588, 1997.

[6] B. Zadrozny and C. Elkan, "Obtaining calibrated probability estimates from decision trees and naive bayesian classifiers," in *In Proceedings of the Eighteenth International Conference on Machine Learning*, 2001.

[7] H. Bostrom, "Calibrating random forests," in *Machine Learning and Applications, 2008. ICMLA'08. Seventh International Conference on*, pp. 121–126, IEEE, 2008.

[8] A. Niculescu-Mizil and R. Caruana, "Obtaining calibrated probabilities from boosting.," in *UAI*, 2005.

[9] Y. Freund and R. E. Schapire, "Experiments with a new boosting algorithm," in *Machine Learning, Proceedings of the Thirteenth International Conference on*, pp. 148–156, IEEE, 1996.

[10] J. C. Platt, "Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods," in *Advances in large margin classifiers*, Citeseer, 1999.

[11] H.-T. Lin, C.-J. Lin, and R. C. Weng, "A note on platt's probabilistic outputs for support vector machines," *Machine learning*, 2007.

[12] S. Schulter, P. Wohlhart, C. Leistner, A. Saffari, P. M. Roth, and H. Bischof, "Alternating decision forests," in *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2013.

[13] S. Bernard, S. Adam, and L. Heutte, "Dynamic random forests," *Pattern Recognition Letters*, vol. 33, no. 12, pp. 1580–1586, 2012.

[14] G. Fan, Z. Wang, and J. Wang, "Cw-ssim kernel based random forest for image classification," in *Visual Communications and Image Processing 2010*, pp. 774425–774425, International Society for Optics and Photonics, 2010.

[15] S. Bernard, L. Heutte, and S. Adam, "On the selection of decision trees in random forests," in *Neural Networks, 2009. IJCNN 2009. International Joint Conference on*, pp. 302–307, IEEE, 2009.

[16] S. Bernard, S. Adam, and L. Heutte, "Using random forests for handwritten digit recognition," in *Document Analysis and Recognition, 2007. ICDAR 2007. Ninth International Conference on*, vol. 2, pp. 1043–1047, IEEE, 2007.

[17] J. Stallkamp, M. Schlipsing, J. Salmen, and C. Igel, "Man vs. computer: Benchmarking machine learning algorithms for traffic sign recognition," *Neural Networks*, no. 0, pp. –, 2012.

[18] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[19] K. Bache and M. Lichman, "UCI machine learning repository," 2013.

[20] J. Hull, "A database for handwritten text recognition research," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 16, pp. 550–554, May 1994.

[21] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning*. Springer Series in Statistics, New York, NY, USA: Springer New York Inc., 2001.