Evaluation of Optimum Path Forest Classifier for Pedestrian Detection

Wendell Fioravante Silva Diniz, Vincent Frémont, Isabelle Fantoni, Euripedes Nobrega

Evaluation of Optimum Path Forest Classifier for Pedestrian Detection*

Wendell F. S. Diniz[1,2], Vincent Fremont[2], Isabelle Fantoni[2], Eurípedes G. O. Nóbrega[1]

***Abstract*— Machine learning (ML) and image processing techniques have been applied together to various scenarios for the development of Intelligent Vehicles. Among these scenarios, pedestrian detection has received growing interest in recent years, since high concern for safety applications in traffic has arisen. Several ML methods were successfully applied to solve this problem. However, because pedestrian detection is in general computationally intensive, a good trade off between accuracy and processing time is desirable, particularly if the methods are directed to real-time applications. Optimum Path Forest (OPF) classifier is a recently developed non-parametric classifier method. This work contribution is the performance assessment of a novel OPF application to pedestrian detection. Results have shown that it is fast and competitive against established methods and a viable alternative to be considered for machine learning and pedestrian detection applications.**

I. Introduction

Classifiers are a very important tool for most of the machine learning applications used to perform many tasks in Intelligent Vehicles field. A classifier is a model-based algorithm that follows given rules to attribute a set of data to some class. In general, a series of values are extracted from the data elements in the form of a feature vector to accomplish the classification goal. Computer Vision techniques are often used to provide the feature vectors. Combining computer vision based feature extraction with a machine learning method, a variety of tasks can be achieved. This framework enables applications such as object detection for driver assistance or autonomous navigation. In the driver assistance field, efficient pedestrian detection is of primary interest to improve traffic safety and increasing safe-guarding of human lives. It can be used in applications as showed by [1], to prioritise defensive driving actions or warn the driver of pedestrian presence.

Among the different feature extraction methods proposed in the field, the Histogram of Oriented Gradients (HOG) and its variations stand out as the most used methods. The combination of HOG features with a machine learning method was successfully applied to the field. Examples of this approach include [2], [3], where the combination of a linear Support Vector Machines (SVM) with HOG has demonstrated to significantly outperform previous methods used for human detection, [4], [5], demonstrating the use of Artificial Neural Networks (ANN) as Multi-layer Perceptrons (MLP) and Random Forests were used in [6]. Many other methods and benchmarking strategies are listed in [7].

Recently, a new classifier based in Graph Theory, the Optimum Path Forest (OPF) [8], has been presented and was successfully applied on different problems, e.g. disease identification [9], and land use estimation used to forest monitoring [10]. These works have shown that OPF usually achieve performances similar to the SVM, but with faster processing times during the classification stage. Because OPF is a parameter-free algorithm, it leads in general to a simpler training routine. These characteristics make OPF a suitable candidate to be used with computer vision techniques in scenarios that require fast performance, like real-time pedestrian detection.

This work contribution is the performance assessment of OPF classifier applied to pedestrian detection. As this field is a sensible area where good quality results are highly desirable, the results achieved by OPF in other fields justify an evaluation of its suitability for pedestrian detection, considering that it was never yet applied to that. Moreover, its suitability to embedded implementation in specialised programmable hardware device is also an important aspect to consider in a future step of this research. Considering all this arguments, we propose to evaluate pedestrian detection based on OPF, through a performance comparison with common methods solving the problem in similar way, namely SVM, MLP and Random Forest. The classifier input data comes from a HOG feature extraction. Impact in the processing time and classification performance caused by a dimension reduction technique is also evaluated for all methods. Profiling was done using canonical metrics, providing a mean for their comparison. Public collections of pedestrian and non-pedestrian images provide the dataset to analyse the method's performances.

The paper is organised as follows: in the next section, the OPF classifier is presented and its principles explained. Following, Section III presents the application scenario, the methodology used to evaluate their performances and the comparison of the methods. Section IV gives the statistical analysis of the results. Finally, in the Section V the discussion of the results is presented and extensions to this work are proposed.

*This work is supported by the PDSE program of Coordination for Improvement of High Education Personnel under Brazilian Ministry of Education, process n°13077/2013-09 and carried out in the framework of the Labex MS2T, funded by the French Government through the program "Investments for the future" managed by the National Agency for Research (Reference ANR-11-IDEX-0004-02).

[1]Department of Computational Mechanics, Faculty of Mechanical Engineering, Campinas State University. Rua Mendeleyev, 200 - CEP 13083-860 Cidade Universitária Zeferino Vaz - Campinas - SP E-mail: wfiorava@hds.utc.fr, wdiniz@fem.unicamp.br

[2]Sorbonne universités, Université de technologie de Compiègne, CNRS 7253 UMR/Heudiasyc. CS 60319, 60203 Compiègne cedex

II. Optimum Path Forest classifier

The use of optimum path forest as a classifier derived for its application in image processing as the Image Foresting Transform (IFT), presented in [11]. The IFT concept is to consider an image as a graph, in which the nodes are the image's pixels and the edges are determined by an adjacency relation between those pixels. Obeying an application-defined cost function, the algorithm extracts a forest of minimal cost path trees, using a previously defined set of pixels as roots. In its essence, the IFT consists of the Dijkstra's shortest path algorithm with some modifications, namely the ability of using multiple starting nodes and general cost functions. Later, its concepts were extended from image processing to general graphs [8], thus giving birth to the OPF classifier.

The OPF based classifier consists in evaluating the dissimilarity between the unknown samples and the training samples using this graph theory based approach. It was proposed in the late 2000s and presents interesting characteristics: it is non-parametric, fast, non-complex implementation, intrinsically multi-class, does not make any assumption about the shapes of the classes and can handle some degree of overlapping between classes. The method can be used for both Supervised and Unsupervised Learning variations. We will focus on the Supervised version, as this approach is more suitable to apply in the pedestrian detection task. The implementation used was that of libOPF [12], made by the method's authors themselves.

A. Training stage

Supervised learning classifiers are constructed based on already known samples of the problem. This is called the fitting or training stage. OPF classifiers are based on a Graph Theory approach, thus it does not depend on the shape of the feature space and it is able to handle overlapping without dimension transformations like those that SVM does.

The basic training routine, as presented in [8], is done by presenting a set of samples to the classifier, called the training set. Assuming that the training set has samples of all possible classes, the training starts by constructing a complete graph using the training set samples as nodes. The weights of the edges are given by a dissimilarity function. The Euclidean Distance is the most used one, but it can be substituted by other functions according to specific needs. The Minimum Spanning Tree (MST) is then computed from the complete graph. The MST holds a relation between the nodes; as they will be connected by minimum cost edges, similar samples tend to be strongly connected. The algorithm will associate a path cost to each node and also mark special nodes called prototypes. The prototypes are defined as nodes of different classes that share an edge. That edge is removed and the prototype associated costs set to 0. The algorithm follows as shown in Fig. 1.

By removing the edges between prototypes, the MST is partitioned into a collection of trees, thus an Optimum Path Forest. Each tree is rooted in a prototype, with all the samples in a given tree belonging to the same class. Each node has

Require: Training set T, prototypes set $S \subset T$
Auxiliary: priority queue Q, real variable cst
Output: classifier P, cost map C, label map L

1: **function** OPF_Training(T)
2: **for all** $s \in S$ **do**
3: $C(s) \leftarrow 0$, $L(s) = \lambda(s)$, insert s in Q
4: **end for**
5: **while** Q is not empty **do**
6: Remove s from Q so that $C(s)$ is minimal
7: **for all** $t \in T | t \neq s$ and $C(t) > C(s)$ **do**
8: $cst \leftarrow \max\{C(s), d(s,t)\}$
9: **if** $cst < C(t)$ **then**
10: **if** $C(t) \neq \infty$ **then**
11: Remove t from Q
12: **end if**
13: $P(t) \leftarrow s$, $L(t) \leftarrow L(s)$, $C(t) \leftarrow cst$
14: Insert t in Q
15: **end if**
16: **end for**
17: **end while**
18: **end function**

Fig. 1: The OPF classifier training algorithm. The priority queue is a special data structure that has a ordered storage policy, it ensures that the element in the head has always the minimum cost among the elements in the queue and also permits arbitrary removal.

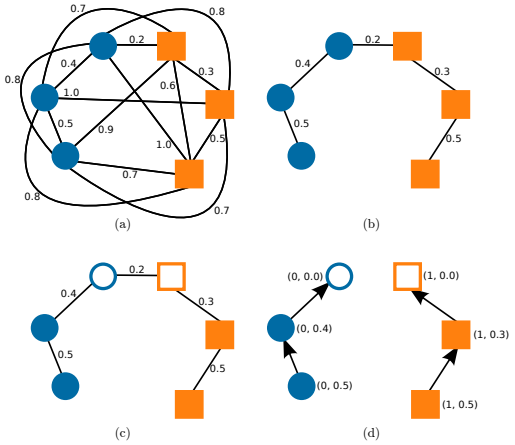


Fig. 2: Training sequence for the OPF classifier. (a) The complete graph with edges weighed by dissimilarity. (b) The Minimum Spanning Tree is found. (c) The next step is to mark the prototypes and associate the costs of each node. (d) With the separation of the prototypes and costs assigned to each node, we have an Optimum Path Forest classifier.

their associated path cost given by the maximum arc weight in the path to its corresponding prototype. Fig. 2 illustrates the classifier construction.

The resulting forest can be directly used to classify the unknown samples, but some methods were proposed to increase the accuracy of the classifier [8], [13] by using another set of samples called the evaluation set. An OPF classifier is built from the training set and its accuracy is evaluated by classifying the samples in the evaluation set. A learning procedure is applied, consisting in switching misclassified nodes in the evaluation set with randomly chosen samples of the training set, reapeting the procedure with the new sets. After a number of iterations, the best instance is selected as the classifier.

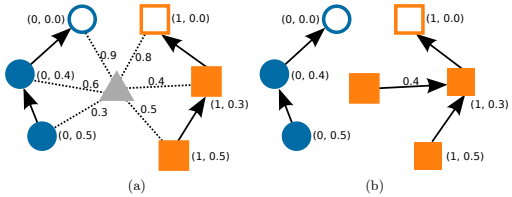


Fig. 3: Classification sequence for the Optimum Path Forest classifier. (a) The unknown sample is presented to the classifier nodes. They offer it their weights to connect to the sample. (b) The node that offers the minimum cost path, considering the cost function presented in (1), connects to the unknown sample and gives it its class label. Note that although the closer node to the unknown sample is of the class circle, the more connected prototype is of the class square, thus the sample is classified as square.

B. Classification stage

The resulting class $\lambda(s)$ of a unknown sample s will be giving by the function below:

$$\lambda(s) = \min_{\forall t \in T}\{\max\{C(t), d(s,t)\}\}, \quad (1)$$

where T is the classifier, $C(t)$ is the associated cost of the classifier node and $d(s,t)$ is the distance between the samples in testing. This means that the sample will be classified as belonging to the same class as its more connected prototype, as illustrated in Fig. 3.

Although the name similarity, an Optimum Path Forest classifies the data in a different method than a Random Forest. In Random Forests, each tree is a decision tree, whose nodes are split paths based in the samples' features. In OPF, each tree is a collection of samples belonging to a common class, with a representative sample acting as root and located near a decision border. It means that in OPF, the classification is done by similarity, consisting in finding the cluster that offers the minimum dissimilarity to connect with the unknown sample.

One key feature of the OPF classifier in comparison with SVM, MLP and Random Forests is that it has no parameters to set, than there is no need to an extra step to get a good classifier. With SVM, often a parameter optimization with k-fold sampling must be done to ensure a good classifier; Random Forest also has some parameters that can influence the performance, like tree depth, number of tree and randomness factor [14]. Finally, with MLP, determining the number of neurons in the hidden layer and the number of hidden layers itself is not trivial [15].

III. Application Scenario and Evaluation Methodology

In order to evaluate the classifiers in a specific scenario that is widely used in the intelligent vehicles field, the pedestrian detection problem was chosen. The samples were composed by mixing various publicly available datasets. They are the INRIA Person [1] dataset, the TUD-Motion Pairs[2] dataset, the Caltech Pedestrian Detection[3] dataset and the CVC-01 Pedestrian[4] dataset. These datasets present little differences in their pose extraction and bounding that are expected to increase the classifiers generalization. The final dataset consisted in 6,080 pedestrian and 6,080 non-pedestrian images. The images are in original scale, with 128×64 pixels resolution, to match the detection window, in PNG format in grey-scale. The Fig. 4 shows a sample of each class. The classifiers were modelled to perform a binary classification, pedestrian images being considered as positive cases and the non-pedestrian as negative cases.

(a) (b)

Fig. 4: Sample of images from the mixed dataset used in this work. (a) A positive sample, showing a person in an arbitrary pose. (b) A negative sample, with no person visible.

A. Feature Extraction

The size of the feature vector directly impacts on the classifier's performance and the quality of the scene analysis. In this paper we use HOG features to extract the main characteristics of the object under analysis, a common feature extraction method for pedestrian detection [2]. The implementation uses the OpenCV library, using the following setup: the detection window is divided into a 8×16 grid, with each cell measuring 8×8 pixels and being non-overlapping. Four neighbouring cells form a block, that will overlap by one cell in horizontal and vertical directions, resulting in $7 \times 15 = 105$ blocks in a window. For each cell, the histogram is divided in 9 bins for the gradient angle, each bin will accumulate the gradient's magnitude. The histograms are concatenated, with each block resulting in $4 \times 9 = 36$ elements. After concatenating all the blocks, our HOG feature vector will be given, resulting in $105 \times 36 = 3780$ dimensions.

As the HOG results in high dimensional feature vectors, it can be useful to apply a dimension reduction method, like Principal Component Analysis (PCA), allowing faster classification times. This time reduction is a desirable effect for real-time applications. A comparison of the methods performances in different PCA configurations was made using the PCA implementation found in OpenCV library [16].

B. Methods for Comparison

The performance of OPF was compared with SVM, MLP and Random Forest with the configurations as follows.

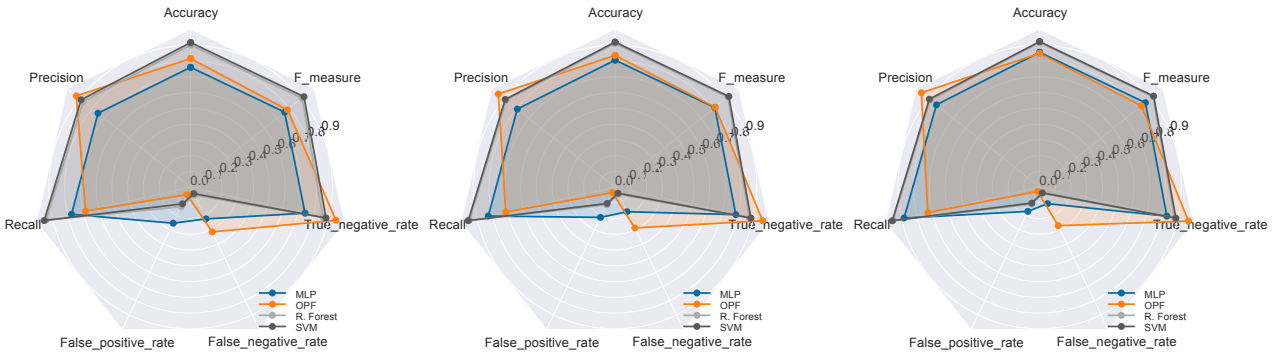
The SVM implementation used was the one from libSVM library [17]. The kernel used was the linear one, as this

[1] At: http://pascal.inrialpes.fr/data/human/

[2] At: http://datasets.d2.mpi-inf.mpg.de/tud-brussels/tud-brussels-motionpairs.tar.gz

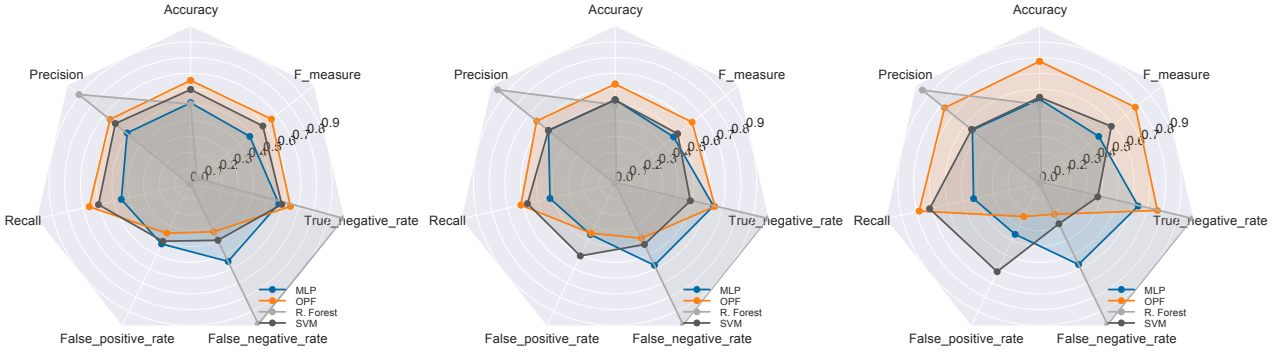
[3] At: http://www.vision.caltech.edu/Image_Datasets/CaltechPedestrians/

[4] At: http://www.cvc.uab.es/adas/site/?q=node/7



(a) 10% samples (b) 25% samples (c) 50% samples

Fig. 5: Metrics for classifying HOG descriptors.



(a) 10% samples (b) 25% samples (c) 50% samples

Fig. 6: Metrics for classifying HOG+PCA descriptors.

TABLE I: Training and testing stages processing times using only HOG descriptors.

% of Samples	Method	Training Time (s)	Testing Time (s)
10%	MLP	115.055	1.109
	OPF	2.910	0.565
	R. Forest	10.530	0.010
	SVM	0.859	0.613
25%	MLP	134.022	2.953
	OPF	19.438	3.594
	R. Forest	40.464	0.040
	SVM	5.497	3.807
50%	MLP	394.981	5.436
	OPF	79.235	13.657
	R. Forest	91.512	0.099
	SVM	20.360	13.887
75%	MLP	791.723	8.506
	OPF	182.043	31.097
	R. Forest	1,546.566	0.164
	SVM	47.600	32.271

TABLE II: Training and testing stages processing times for PCA+HOG descriptors.

% of Samples	Method	Training Time (s)	Testing Time (s)
10%	MLP	22.521	0.176
	OPF	0.728	0.143
	R. Forest	2.415	0.005
	SVM	4.539	0.118
25%	MLP	169.328	0.686
	OPF	8.156	1.468
	R. Forest	10.199	0.017
	SVM	32.166	1.065
50%	MLP	689.363	1.643
	OPF	49.831	7.184
	R. Forest	21.251	0.041
	SVM	97.898	4.731
75%	MLP	1,168.172	2.578
	OPF	111.896	18.584
	R. Forest	39.169	0.068
	SVM	164.854	10.919

type of kernel usually selected to be used in pedestrian detection [2], [18].

The MLP and Random Forest implementations used were the ones from OpenCV library. MLP used the RPROP (Resilient Propagation) [19] method of training. Random Forest was set to have a maximum number of 100 forests and maximum depth of 25. The OpenCV version used was built with Intel™ Thread Building Blocks library. As stated in OpenCV's documentation, both MLP and Random Forest implementations benefit from the library's parallelization. This

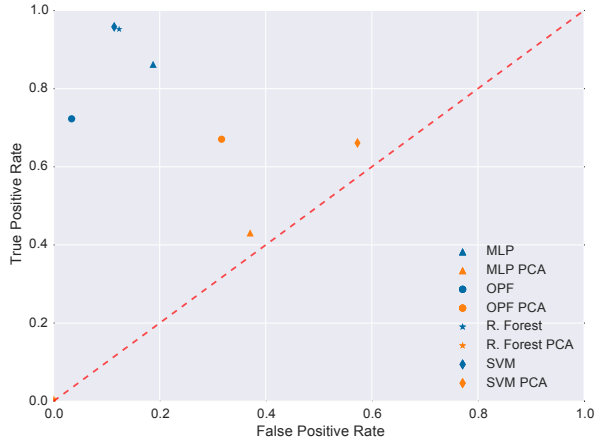


Fig. 7: Receiver Operating Characteristic space results for each descriptor. Best viewed in color.

has to be considered for the processing times comparison, as the other methods' implementations do not use any kind of parallelism.

Only the classifier performance was evaluated, using a Per Window approach, as described in [20]. The intention is evaluate the suitability of the classifier to be applied to Pedestrian Detection applications. The final efficiency of the classifier part is greatly influenced by the detector part, but evaluating the classifier alone give us base to chose the more suitable for the applied scenario, let say, focusing in classification performance or processing speed.

IV. Experimental Results

The metrics used to evaluate the method's performances were the ones based in the Confusion Matrix. The time spent in training and testing phases have also been considered. They were measured using Repeated Random Sub-sampling validation with stratified sampling, keeping $50-50$ ratio between positive and negative samples in every partition. Each method was executed 100 times with a different randomly chosen set of training and test samples keeping the same sets for each method in each round. The final results are given by the arithmetic mean of the rounds. The equipment used was a PC equipped with a Intel® Core™ i7-3720QM CPU at 2,6 GHz with 8 GB RAM DDR2 running Ubuntu 14.04 "Trusty Tahr".

All classifiers were evaluated with HOG alone and HOG+PCA features. In order to evaluate the influence of the number of samples on classification stability, four configurations were used, with 10%, 25%, 50% and 75% of the dataset respectively. An amount of 40% of the resulting set was used to train the classifiers and other 40% used as test set. The remaining 20% were used for the training of the OPF method, as it require an extra validation set.

The execution time for applying PCA and projecting the samples to the resulting subspace was, in average, 32 minutes. We chose to keep 95% of the original covariance, reducing the feature space from 3780 to 1271 dimensions.

Fig. 5 shows the results for HOG descriptors and Fig. 6 shows the results for HOG+PCA descriptors. As the result for 75% were practically the same as for 50%, we suppressed that result. For HOG only, SVM and Random Forests had practically the same performance, with OPF and MLP being a little less accurate. With HOG+PCA, all methods showed a drop in performance. We can notice that the OPF method was less affected, showing to be stable and more accurate than the other methods in this configuration. The MLP was stable but significantly less accurate and Random Forest completely degenerates.

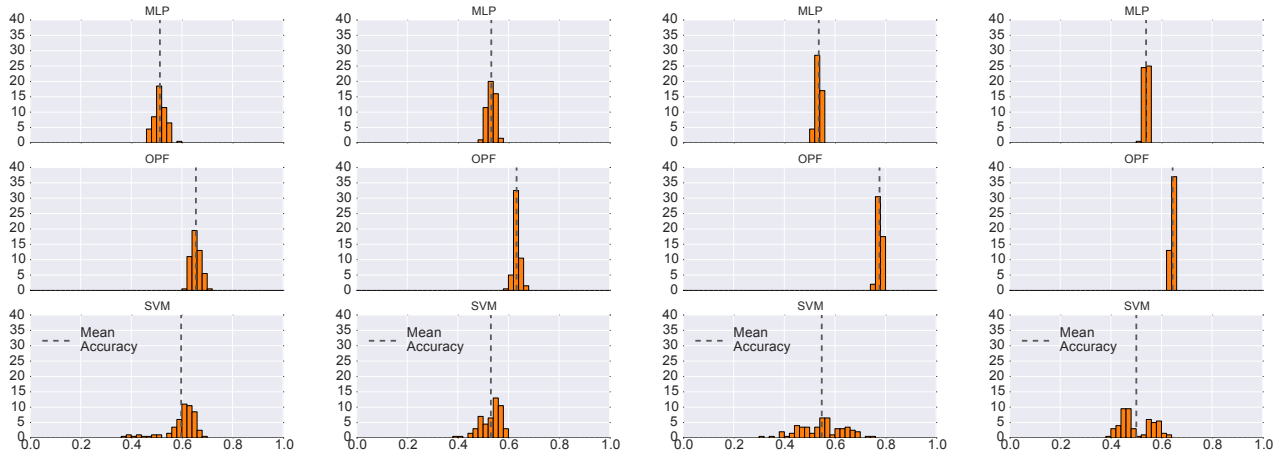
Fig. 7 shows the resulting Receiver Operating Characteristic space using each descriptor. As the OPF is a discrete classifier, the resulting ROC curve is a single point. To be fair, all the other methods were also set as discrete. Table I and Table II show the processing times for training and testing stages. We can notice a significant reduction in testing time with HOG+PCA and an increase in training time, except for OPF, whose training time decreased in all situations. Random Forest showed unmatched speed, being the faster with HOG descriptors, but given its poor performance with HOG+PCA, the results for this feature extraction method must be disregarded. OPF and SVM showed close speed results, with OPF being more accurate with HOG+PCA. When the number of samples is increased, the advantage of the TBB library parallelization in OpenCV methods is noticed; MLP and R. Forest became faster. It is also important to remark that the parameter optimization performed by the libSVM with HOG+PCA had some difficulty to converge. This can be an indicative that within HOG+PCA subspace, the linear kernel lost its generalization, bringing the necessity of testing different kernels or doing a deeper parameter optimization. This remarks the advantage of the non-parametric characteristic of the OPF, alongside its stability with dimension reduction by HOG+PCA.

Fig. 8 shows the Accuracy histogram of each method for HOG+PCA descriptors. OPF shows to be more stable and accurate. We have to disregard Random Forest result, as it is not functional with this descriptor.

Although other metrics have shown good results, the false negative rate for OPF is a bit higher than expected, with values around 20%. As false negative means that the presence of a pedestrian was not detected, this metric in particular is important to improve the safety and efficiency of the system. One can say that this is the most important feature expected in a pedestrian detection system.

V. Conclusions

A novel application of the OPF classifier for pedestrian detection using HOG feature extraction alone and with PCA dimension reduction is analysed. We compared its performance with other methods usually applied for this task. It is important to notice that dimension reduction done by PCA significantly influences the methods' performances, with OPF showing to be less sensible. Therefore, it is possible to take advantage of the reduction in processing time without compromising too much of the classification performance. Its simplicity of implementation and absence of parameters in training routine and multi-class capability make it a suitable



(a) 10% samples (b) 25% samples (c) 50% samples (d) 75% samples

Fig. 8: Accuracy histogram showing classification stability with 100 randomly chosen training sets for each method using HOG+PCA descriptors. Random Forest results was disregarded, as it is not functional.

candidate for its use in pedestrian detection applications. OPF algorithm also shows great potential for parallelism, being suitable to implementation in specialized hardware like GPUs or FPGAs, which will permit applications in real-time embedded systems.

Future extensions of this work will consider applying a feature selection method alongside PCA, in order to improve accuracy and false negative rate. Furthermore, investigating other dimension reductions techniques may help to improve the classifier's performance. As the detector performance affects the results, a further extension will re-evaluate the OPF classifier with a complete system, thus permitting per frame evaluation, a more commonly found benchmarking for Pedestrian Detection systems, as well permitting the use of standard datasets.

REFERENCES

[1] F. Rodriguez and V. Fremont, "An embedded multi-modal system for object localization and tracking," *Intell. Transp. Syst. Mag. IEEE*, vol. 4, pp. 1–11, 2012.

[2] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," *2005 IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, vol. 1, pp. 886–893, 2005.

[3] G. Zhang, F. Gao, C. Liu, and W. Liu, "A pedestrian detection method based on svm classifier and optimized histograms of oriented gradients feature," in *2010 Sixth Int. Conf. Nat. Comput.*, vol. 6, aug 2010, pp. 3257–3260.

[4] L. Zhao and C. Thorpe, "Stereo- and neural network-based pedestrian detection," *IEEE Trans. Intell. Transp. Syst.*, vol. 1, pp. 148–154, 2000.

[5] M. Szarvas, A. Yoshizawa, M. Yamamoto, and J. Ogata, "Pedestrian detection with convolutional neural networks," in *IEEE Proceedings. Intell. Veh. Symp. 2005.*, 2005, pp. 224–229.

[6] J. Marin, D. Vazquez, A. M. Lopez, J. Amores, and B. Leibe, "Random forests of local experts for pedestrian detection," in *2013 IEEE Int. Conf. Comput. Vis.*, dec 2013, pp. 2592–2599.

[7] R. Benenson, O. Mohamed, J. Hosang, and B. Schiele, "Ten years of pedestrian detection, what have we learned?" in *Eur. Conf. Comput. Vis. - ECCV*, 2014.

[8] J. P. Papa, A. X. Falcão, and C. T. N. Suzuki, "Supervised pattern classification based on optimum-path forest," *Int. J. Imaging Syst. Technol.*, vol. 19, pp. 120–131, jun 2009.

[9] A. A. Spadotto, J. C. Pereira, R. C. Guido, J. P. Papa, A. X. Falcao, A. R. Gatto, P. C. Cola, and A. O. Schelp, "Oropharyngeal dysphagia identification using wavelets and optimum path forest," in *2008 3rd Int. Symp. Commun. Control Signal Process.*, mar 2008, pp. 735–740.

[10] R. Pisani and P. Riedel, "Land use image classification through optimum-path forest clustering," *Geosci. Remote Sens. Symp. (IGARSS), 2011 IEEE Int.*, pp. 826–829, jul 2011.

[11] A. X. Falcão, J. Stolfi, and R. de Alencar Lotufo, "The image foresting transform: Theory, algorithms, and applications," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 26, pp. 19–29, jan 2004.

[12] J. Papa, C. Suzuki, and A. Falcao, "Libopf a library for the design of optimum path forest classifiers," 2014.

[13] J. P. Papa, F. a.M. Cappabianco, and A. X. Falcao, "Optimizing optimum-path forest classification for huge datasets," in *2010 20th Int. Conf. Pattern Recognit.*, aug 2010, pp. 4162–4165.

[14] A. Criminisi, J. Shotton, and E. Konukoglu, "Decision forests for classification , regression , density estimation , manifold learning and semi-supervised learning," Tech. Rep., 2011.

[15] K. G. Sheela and S. N. Deepa, "Review on methods to fix number of hidden neurons in neural networks," *Math. Probl. Eng.*, vol. 2013, 2013.

[16] G. Bradski, "{The OpenCV Library}," *Dr. Dobbs J. Softw. Tools*, 2000.

[17] C.-C. Chang and C.-J. Lin, "{LIBSVM}: A library for support vector machines," *ACM Trans. Intell. Syst. Technol.*, vol. 2, pp. 27:1—-27:27, 2011.

[18] T. Kobayashi, A. Hidaka, and T. Kurita, "Selection of histograms of oriented gradients features for pedestrian detection," in *Neural Inf. Process.*, 4985th ed., ser. Lecture Notes in Computer Science, M. Ishikawa, K. Doya, H. Miyamoto, and T. Yamakawa, Eds., Berlin, Heidelberg, 2008, vol. 4985, pp. 598–607.

[19] M. Riedmiller and H. Braun, "A direct adaptive method for faster backpropagation learning: the rprop algorithm," in *Neural Networks, 1993., IEEE Int. Conf.*, vol. 1, 1993, pp. 586–591.

[20] P. Dollár, C. Wojek, B. Schiele, and P. Perona, "Pedestrian detection: an evaluation of the state of the art." *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 34, pp. 743–61, apr 2012.