Discriminative tree-based feature mapping

Miroslav Kobetski
http://www.csc.kth.se/~kobetski/
Josephine Sullivan
http://www.csc.kth.se/~sullivan/

Computer Vision Group, CVAP
KTH, Royal Institute of Technology
Teknikringen 14,
114 28 Stockholm

Abstract

For object classification and detection, the algorithm pipeline often involves classifying feature vectors extracted from image patches. Existing features such as HOG, fail to map the image patches into a space where a linear hyperplane is suitable for separating the classes, while many non-linear classification methods are too expensive for many tasks. We propose a sparse tree-based mapping method that learns a mapping of the feature vector to a space where a linear hyperplane can better separate negative and positive examples. The learned mapping function $\Phi(x)$ results in significant improvement for image patch classification with HOG and LBP-features over other feature mapping methods on VOC2007 and INRIAPerson datasets.

1 Introduction

Some of the best performing image classification methods rely on combinations of non-linear kernels to achieve good results [18]. Such methods are computationally too slow for many problems. They are not feasible for detection unless combined with cascades of simpler methods, and even then they are prohibitively slow for some tasks. To overcome computational in-feasibility while still achieving good results, linear models are often enriched with parts[5] or combinations of various features or mappings [12, 16, 20, 21].

Good features $x(I)$ can be seen as functions that map from pixel intensities I to a feature space where classes are more easily separable. Parts can then be thought of as an additional layer $\Phi(x,z)$ where the mapping function is also parametrised by part position z. Non-linear kernels represent another layer that can be seen as implicitly mapping x into yet another (potentially non-finite) space where the classes may or may not be more easily separable, depending on the properties of the kernel. A number of non-linear kernel have been made feasible for a larger number of problems, via finite approximations [10, 12, 16]. Although it is very useful to have a feature mapping approximating a certain kernel, it is often difficult to directly translate this to better discriminative performance for a particular problem, class and feature. Also, many powerful kernels still lack good low-dimensional approximations.

Tree-based classifiers such as boosted ensembles or random forests are powerful and fast, but are not easy to integrate with other well-developed methods that rely on explicit feature representations. In this paper we propose a more direct approach for finding a low-dimensional feature mapping $\Phi(x)$ to a space where the object and background classes are more easily separated by a linear hyperplane. Rather than finding a kernel with these properties and approximating it, we directly learn a non-linear decision boundary using boosted decision trees. We then induce the new feature dimensions from the decision paths or rules of the tree. We propose two methods for inducing $\Phi(x)$ from trees. We show that they both increase discriminative performance compared to our baseline, to a number of competing methods at a small cost of evaluating the $\Phi(x)$ function. Linear SVMs in the tree-mapped space even outperform the original tree-based classifiers.

2 Related work

Improving the discriminative performance of methods that are feasible for detection is a busy research field. Dalal and Triggs' HOG feature [2] has been extended by separating the single-template HOG model into a number of mixture components and by adding parts, greatly improving detection performance [5]. It has later been shown that somewhat similar performance can be obtained without the parts if the visual similarity measure used to cluster examples is more competent [1, 3]. Exemplar-SVMs can be seen as an extreme case of this, where each example is associated with one linear decision boundary [13]. As these methods are all based on linear filters and HOG features, we believe that they can be used together with our method, since the main output from our algorithm is a feature mapping $\Phi(x)$ that can be used as an intermediate step after computing the feature vector.

Feature mappings can also be induced as approximations of kernels [16]. Approximating a stationary or homogeneous kernel with a finite-dimensional feature map $\Phi(x)$ to use as input to a linear classifier improves classification performance significantly at the cost of increased feature dimensionality. A sparse approximation applied to detection has produced a 32 times sparser HOG feature, at a small performance loss [17]. The additive classifiers of Maji [10, 11, 12] also have feature mapping interpretations, and are also outperforming linear classifiers by approximating kernels. Our work differs from the above works in that we do not try to approximate a specific kernel, that may or may not have characteristics that are beneficial for classification for a certain problem, feature or class. Instead we directly learn a feature mapping that transforms the original feature to a space where a linear classifier is better able to separate the examples. The margin-maximizing property of the boosted tree classifier, from which we induce our features, ensures that this linear separablity translates to improved discriminative ability also on unseen test data.

Inducing binary rules from a random forest has previously been proposed and demonstrated on typical machine learning datasets [19]. Our methods for inducing features preserve the score of the different decisions in the decision path and allow for richer feature dimensions. We also think that the greedy nature of boosted trees is more suitable for inducing feature dimensions in vision, as we often deal with high-dimensional spaces and many of the original features in x are likely to be non-discriminative or noisy. Rulefit is a method that does not explicitly create a feature mapping, but that looks for sparse combinations of tree rules from a pool of tree based classifiers [6]. The main difference is that Rulefit does not explicitly define a feature mapping $\Phi(x)$, so in order to induce dimensions from the learnt rules one would still have to apply one of our algorithms.

As Vens points out [19], tree-based feature induction also has some similarities to distance metric learning [22] although the flexibility of a tree based mapping is much higher than typical distance-metric-learnt mappings.

3 Method

Rather than formulating an explicit mapping for a kernel function that may or may not be appropriate, we construct a mapping by inducing dimensions from the rules of a learnt tree-based classifier.

Trees are powerful classifiers that are able to define highly non-linear decision boundaries. Even with good learning heuristics, decision trees have little or no notion of margin and will overfit to the data, and to avoid this we base our feature mapping on a boosted sequence of trees. This counteracts overfitting and has been shown to have margin maximising properties [7]. Altough a final SVM layer will achieve margin maximisation in the mapped space, we believe that it is important that trees from which the features are induced already have a wide margin. If the tree-based classifier has overfit, our induced feature mapping will result in a space where the linear separability of training examples does not translate well to linear separability of unseen test examples.

One main reason for inducing a feature mapping rather than using the trees directly is that many existing methods and frameworks (many generative models, clustering methods, detection frameworks) require an explicit feature representation. Hard negative mining is another example of a very important technique that is well defined for linear SVMs [5], but more of an unstable art-form for tree ensembles.

3.1 Mapping the tree by adding dimensions

Each feature dimension $z_j = \phi_j(x)$ of our mapped feature vector $z = \Phi(x)$ encodes the decision path S_j from the root to one leaf node of the tree, also known as a rule. Starting at the root, the path to a leaf node can be seen as a sequence of decisions based on a number of feature values x_i and thresholds α_i, where x_i is the i:th dimension of feature vector x. For example, a three-level tree might have a path to node j that looks like $S_j(x) = (x_{j_1} > \alpha_{j_1}) \wedge (x_{j_2} < \alpha_{j_2}) \wedge \left(x_{j_3} > \alpha_{j_3}\right)$. We define

$$S_j(x) = \prod_{k=1}^{n} \delta_{jk}(x_{j_k}, \alpha_{j_k}), \tag{1}$$

where $\delta_{jk} \in \{\delta^+, \delta^-\}$ encode binary decisions $\delta^+(z_1, z_2) = [z_1 \geq z_2]$ and $\delta^-(z_1, z_2) = [z_1 < z_2]$, described using Iverson bracket notation. Such rules are inherently binary, so we enrich our feature mappings by also considering distance to the decision boundary or rule margin $m_j(x)$. Introducing the concept of rule margin adds some ambiguity to the feature induction, since for a rule j, $m_j(x)$ is the accumulated score of its decisions and that accumulation can be envisioned and implemented in a number of ways. The representation of $m_j(x)$ is based on how one chooses to consider the dependence between decisions and the properties one wants the feature mapping to have. We present two of our different tree-mapping algorithms and show how their conceptual dissimilarities result in quite different decision boundaries and results for different problems.

3.2 Mapping algorithms

3.2.1 LDM - Leaf node decision mapping

Leaf node decision mapping is the most straight-forward addition of $m_j(x)$ to the otherwise discrete tree mapping. For a leaf node of depth n, the mapping simply becomes

$$\phi_j(x) = S_j(x)(x_{j_n} - \alpha_{j_n}). \tag{2}$$

The non-linearity is encoded through the binary decisions made following the path $S_j(x)$. An example receives zero score if it fails any of the binary decisions. If all decisions are true the example gets a value signifying its signed distance to the decision threshold made just prior to the leaf node. Each added dimension can be seen as a partition of space where similar examples are ordered according to feature x_{j_n} and dissimilar examples (as they have failed earlier similarity tests) are ignored. Due to this partitioning property LDM closely resembles the decision rules from which it is induced.

3.2.2 PCM - Polynomial cross-term mapping

The margin of an example is often used as a score representing the certainty of its classification. In the case of LDM all information about the distance to each splits except the final one is lost. One could try to retain this information by accumulating the score from different splits along S_j.

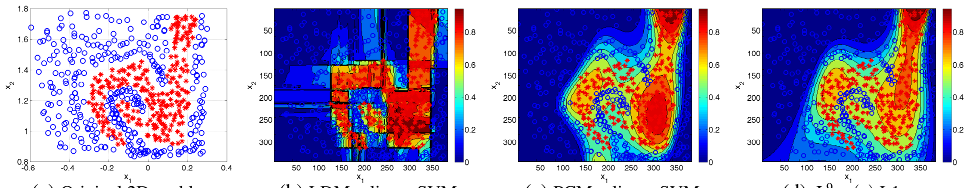
$$\phi_j(x) = \prod_{k=1}^{n}(2\delta_{j_k} - 1)(x_{j_k} - \alpha_{j_k}). \tag{3}$$

This can be seen as selecting a subset of all cross terms of a polynomial kernel, where the depth of the tree determines the degree of the polynomial. Since the selected features are all scaled around different decision thresholds this mapping has additional flexibility to that of a polynomial kernel. It is also much sparser since only the cross terms of features that jointly produce discriminative decisions for the particular class are included.

3.3 Illustrative toy example

Figure 1a shows a 2-D toy example with a complicated boundary between the positive and negative class. A linear decision boundary applied directly to the 2D data could obviously not do a good job here, while a boosted tree-classifier would have no problem defining a jagged but reasonable decision boundary by combining decision trees.

In the images seen in figure 1 a linear SVM's decision boundary is mapped back down to the 2-D example for a number of different feature mappings. We see that a hyperplane is able to separate the negatives from positives quite well in the space of our proposed mapping methods. While the linear decision boundary in LDM-mapped space resembles the sharp decision boundary of a boosted tree classifier, the PCM-mapped line looks much less like a typical tree and more like a polynomial kernel. An L1-regularised linear SVM with a high-degree polynomial mapping achieves a decision boundary that is similar to PCM. In this 2-dimensional toy problem it is possible to enumerate all the polynomial cross-terms to create the polynomial mapping; in real problems this is infeasible as $dim(\Phi_{poly}(x)) \propto D^d$, for a feature of dimension D and a d-degree polynomial mapping. And while a sparse high-degree polynomial kernel does a good job with this problem a χ^2-mapping would not, therefore this contrived example illustrates our point. Successful application of a kernel-approximate



(a) Original 2D problem (b) LDM + linear SVM (c) PCM + linear SVM (d) $\Phi^9_{poly}(x)$ L1-reg

Figure 1: 2D toy problem where the 2D points have been mapped using tree-based feature mapping, after which a linear SVM was learned and projected back to 2D. $\Phi^d_{poly}(x)$ is the feature map of a d-degree polynomial kernel where each dimension is a polynomial term. The color values represent the scores of the linear SVM.

feature mapping relies on first finding or constructing an appropriate kernel for the problem and then being able to approximate it. Our method learns a mapping specifically for the problem.

4 Experiments

The toy problem illustrated that our tree-based mapping algorithms produce spaces in which the data is easier to separate using a hyperplane. The point of our experiments is to investigate how this translates to high-dimensional vision problems and if such spaces actually result in better discrimination on unseen data. Rather than hundreds of labelled examples distributed in two dimensions as in the toy problem, vision problems typically have many more dimensions per labelled example. We focus on a setting that is quite common in object classification and detection - hundreds to thousands of labelled examples, and feature dimensionality of $\sim$1000. In such a high dimensional space it is likely that many dimensions are non-discriminative and a sparse model is suitable. This also justifies the use of a boosted tree classifier, since its forward stagewise fitting algorithm has a regularisation that has similar properties to the L1-penalty of lasso [8].

Decision trees of different depths are able to capture different degrees of interaction between features. Deeper trees are able to capture high order interaction between feature dimensions. However, this makes them weaker at capturing main and low order interactions between features [6]. We therefore experiment with trees of various depths.

We perform image classification and patch classification experiments on Pascal VOC2007, and pedestrian detection on the INRIAPerson dataset.

4.0.1 VOC2007 patch classification

We use Pascal VOC2007 as our the dataset as it is a standard object class dataset, with relatively little bias [15]. It contains 20 objects classes, with instances having a large variety of poses, subtypes and viewpoints. In order to avoid variations introduced by the and implementation details and engineering requirements of a detection framework we focus on patch classification as the main type of experiment. Positive examples have bounding-box annotations, so we crop and warp all positive examples to have the same patch size. We use a modified HOG descriptor [5] to describe each patch. To get negative examples we generate a large number of random bounding boxes that do not overlap with any positive example of that class. We take care to sample negative boxes with realistic sizes, positions and aspect ratios, and handle them exactly as positive examples in order to avoid any significant difference between negatives and positives that could be exploited by an overfitting algorithm. This produces N_c examples for each class where each example i is represented by a feature vector x_i of dimensionality 1024.

One criticism against patch classification for evaluating object recognition performance is that the object patches are well aligned up to bounding box level, which is unrealistic in application scenarios, such as object detection. A good classifier for object detection must handle misaligned instances well. In order to cope with this we introduce a random offset of a few pixels to all parameters of the bounding box, this way generating misaligned and non-centred examples.

For each of the classes we learn a boosted tree classifier with 6000 leaf nodes, which allows us to induce 3000 new dimensions. Although we learn trees with different depths in our different experiments, we keep the number of leaf nodes constant, meaning that the number of trees vary. We then learn a linear SVM on the new mapped features $\Phi(x)$ using an off-the-shelf SVM software [4]. As a reference we also provide SVMs learned on the original feature space using a linear and a polynomial kernel. We also compare to a linear SVM in χ^2-mapped space and a classifier learned using Rulefit [6]. Besides using each of our algorithms we recognize that they have different properties, the most prominent difference being the ability to partition the space versus fully used features. This suggests that they might be somewhat complimentary, so we also try to learn an SVM on the feature space that is given by several tree-based mappings combined.

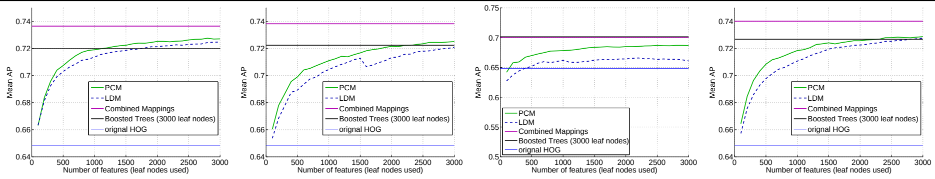
In order to investigate the effects of joint feature interactions of different orders we perform experiments using trees of different depths. We parametrise the depth of the tree by K, the maximum number of splits allowed when learning the tree.

4.0.2 VOC2007 image classification

The image classification task aims to answer if an object is present in an image. This problem is typically approached with variations of Bag-of-words (BoW); we use a a spatial pyramid BoW approach in our experiments [9]. We use a dictionary of 1000 words and 4 spatial bins at the lowest level. The computational bottlenecks of BoW image classification are different from those of detection and kernel methods are actually feasible for some problems. This is a good reference experiment to demonstrate the case when well-adapted and approximable kernels exist. We compare to a Hellinger kernel mapping due to its simplicity and to an approximate χ^2 mapping due to its high performance and appropriateness for the problem.

4.0.3 INRIAPerson classification

We also perform image classification on the INRIAPerson pedestrian dataset following the procedure in [2]. We use both HOG (dimension=3360) and a simpler local binary pattern (LBP) [14] (dimension=7450) feature. INRIAPerson is divided into a training and test set with 2416 and 1132 positive examples respectively. We generate 13000 negative examples for the training, similar to what was used in the original paper. Many pedestrian detection applications require real-time performance, making high-dimensional features less attractive. We therefore only use the induced feature dimensions in this experiment. Since this dataset is considerably easier than VOC2007, the boosted tree performance saturates much sooner, and we only learn a classifier containing 2000 leaf nodes. Using only the induced features the dimensionality is therefore reduced by ≈ 10 times. In this case a χ^2-mapping would



(a) K = 3 (b) K = 7 (c) K = 15 (d) K = {3, 5, 7}

Figure 2: Performance as the number of induced features increases. Tree-based mapping has better relative performance for shallow trees, and already at $K = 15$, it has a difficult time beating the baseline.

result in 75670 feature dimensions (70 times higher than the tree-mapped approach), making it unsuitable from a performance perspective.

5 Results

5.0.4 VOC2007 results

Table 1 shows that our feature mappings significantly outperform the SVM learnt on the original HOG feature, and also perform better than the reference methods. The χ^2 approximation is of third degree as suggested in [16], and that results in 7168 feature dimensions, which is more than twice the dimensionality of our tree-based mappings. The polynomial kernel is provided as reference, although using polynomial mapping is infeasible - degree-3 polynomial mapping would result in $\approx 1.8^9$ dimensions. So looking at PCM as a polynomial mapping with varying feature normalisations, we can see that it is extremely sparse. At 3000 dimensions, figure 2 shows that the performance has not yet saturated and would increase more with additional dimensions.

That PCM and LDM achieve significantly higher discriminative performance than the χ^2-mapping is reasonable since they are learnt with the objective of being discriminative for each particular class. The χ^2 mapping is not particularly suited for this specific task (although it performs best of the additive homogeneous kernel approximations), but other kernels that might be better suited have no known accurate approximation.

Of our algorithms LDM is the one most similar to the binary rules used in Rulefit [6] and [19], and most accurately represents the decisions made by the boosted trees. PCM is the one where $\phi_j(x)$ least faithfully represents the rule from which it was induced, yet produces the highest performance. We believe that this is related to the properties of the linear SVM classifier used as our final layer. Although LDM is closer to the original tree classifier in structure, able to partition the feature space, a linear SVM is not well suited for sparse features, that are zero for most examples.

Figure 2 shows the performance for different tree sizes as the number of induced feature dimensions increases. As previously noted, trees of different depths capture different levels of interaction between features so we see different performance for different K values. This is also evident in figure 2d, where we see that the boosted tree classifier and the induced mapping that mixes trees of different sizes reach the highest performance. Beside capturing more complex joint feature interactions, deeper trees are more likely to overfit, which is probably the reason why $K = 15$ gives much worse performance than the other mappings. It is unclear if clear overfitting starts to occur at tree depth 4, or most of the discriminative feature interactions occur at orders zero to three, and are missed when using deeper trees.

We have argued that our proposed mappings have different properties, where LDM par-

	AP of classifier									
Class	HOG	χ^2-mapped	Rulefit	Polynomial kernel (d=3)	Boosted trees	LDM	PCM	Combined mapping	Improvement using PCM	Improvement using Combined
Aeroplane	66.28	69.51	68.11	70.55	73.10	69.24	**74.70**	73.74	8.42	7.46
Bicycle	86.66	87.28	83.90	86.99	87.61	87.82	86.61	**87.95**	−0.05	1.29
Bird	40.81	42.11	47.78	44.78	51.74	**54.04**	53.85	53.61	13.04	12.80
Boat	42.34	50.23	52.35	54.45	57.60	57.04	58.08	**59.77**	15.74	17.43
Bottle	70.42	71.53	67.60	71.31	73.20	74.24	75.06	**76.81**	4.64	6.39
Bus	81.70	**88.80**	86.87	85.40	87.67	85.05	80.23	84.99	−1.47	3.29
Car	86.63	86.97	89.90	91.34	91.56	90.92	91.66	**92.10**	5.03	5.47
Cat	40.42	45.87	51.15	52.21	58.34	56.55	56.22	**58.36**	15.80	17.94
Chair	60.44	63.28	65.85	68.84	70.55	71.60	70.93	**72.48**	10.49	12.04
Cow	81.42	84.43	77.43	76.13	83.01	84.24	83.97	**85.64**	2.55	4.22
Diningtable	25.33	46.50	39.70	47.40	45.31	48.24	48.85	**50.54**	23.52	25.21
Dog	45.66	48.96	52.03	50.29	59.13	58.56	**60.08**	59.37	14.42	13.71
Horse	74.36	78.30	72.64	73.50	**79.79**	77.53	77.64	78.70	3.28	4.34
Motorbike	73.98	75.68	74.04	78.05	79.10	78.95	77.63	**79.69**	3.65	5.71
Person	57.03	60.24	65.28	75.32	72.85	76.05	77.56	**78.25**	20.53	21.22
Pottedplant	59.94	59.48	58.13	62.96	60.31	62.90	62.61	**63.14**	2.67	3.20
Sheep	84.22	86.96	83.20	82.19	**87.22**	85.66	86.83	86.94	2.61	2.72
Sofa	57.22	**64.52**	57.99	58.08	61.98	63.80	62.46	64.10	5.24	6.88
Train	75.47	83.26	79.41	81.77	83.70	83.35	82.56	**84.44**	7.09	8.97
Tvmonitor	86.71	88.19	88.75	**90.14**	89.83	89.40	89.97	89.60	3.26	2.89
Mean	64.85	69.10	68.11	70.09	72.68	72.76	72.88	**74.01**	8.02	9.16

Table 1: Patch classification performance for a single HOG template and linear SVM with various feature mappings on VOC2007. Boldface values represent the mapping that performs best for each class. The feature mappings are based on a boosted tree classifier that consists of trees with $K \in \{3,5,7\}$, where K is the maximum number of tree-splits. The proposed mappings outperform the other methods, including the boosted tree classifier they were induced from.

titions the space focusing on a number of similar examples, while PCM lacks this ability but instead has more information to be exploited in each dimension. Although PCM is the favourable choice for this setting it does not do as well on the toy problem, which illustrates that the different properties of the mappings can be suitable for different problems. We also see that the mappings are somewhat complementary as combining features from several mappings increases performance even more.

We expected the tree-based mappings to beat the performance of the boosted classifier that was used to induce them, due to the theoretically well-founded max-margin properties of the linear SVM used as a final layer. That we get roughly the same results as the boosted trees should indicate that the margin maximisation property of boosting is quite apt. Rulefit, similarly to an SVM, also searches for a globally optimal rule combination, so we expected it to perform well in comparison, as it had previously outperformed boosted classifiers and random forests on typical machine learning datasets. We argue that boosting can result in better rules than Rulefit for high-dimensional problems. The number of possible feature combinations for the rules is d^n, where d is the feature dimension and n is the tree depth. The high dimensionality of the HOG features makes the space of possible rules much larger than for low-dimensional machine learning problems and it could be that the greedy stepwise learning of a boosted classifier is better able to efficiently explore the huge space of rules.

Results for the VOC2007 image classification experiment are seen in table 2. The tree-based mappings outperform the baseline but do not beat all the reference methods. The χ^2 kernel (and others) is particularly well suited for comparing discrete distributions - here represented as L1-normalized histograms of visual words. This relates to our point that our tree-based approach is a general way of learning a mapping for a given problem, without explicitly considering its properties or having a suitable kernel approximation. When

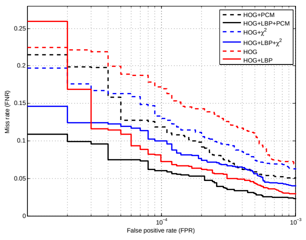


Figure 3: Performance on the INRIA pedestrian dataset **(Lower is better)**. Due to the low FPR numbers, the x-axis has a log scale.

Mapping	mAP	improvement
Linear	42.83	0.00
Hellinger	46.54	3.71
χ^2	47.96	5.13
LDM	46.32	3.49
PCM	43.82	0.99
Combined	47.51	4.68

Table 2: VOC2007 image classification performance. All methods improve performance but only the combined mapping is able to reach χ^2.

a problem-optimal kernel and mapping exists there is little point, other than sparsity and feature selection, for using the more general approach of tree-based mapping.

5.0.5 INRIAPerson results

For the INRIA experiment the standard performance measurement is the *miss rate* vs. *false positive rate*(FPR) seen in figure 3. The main area of interest is in the low-false-positive region, since the intended application is pedestrian detection. A false positive rate of 10^{-4} corresponds to one false positive per image frame if your sliding window stride is $1/100$ of the image size. At FPR 10^{-4} we see that feature mapping reduces the miss rate by by 5.12% (25.65% relative improvement) for the HOG feature and 2.12% (26.06% relative improvement) for the HOG and LBP features combined. Although the figure is zoomed-in on the low-fpr-region, the ordering of the curves is consistent in the higher-fpr-regions, which can be seen in the average precision numbers stated in the figure. Pedestrians in INRIA are well defined by their contour, which HOG describes well. LBP on the other hand is often used for texture description, and each feature dimension is simpler than that of HOG. It is therefore not surprising that the flexibility of the tree-based mapping results in larger improvements for the simpler feature than for the more complex. That the final LBP performance exceeds the performance of HOG could be because that the mapping learnt by the trees creates a more suitable feature than the hand-crafted HOG. Having a simpler and less constrained feature to start from could allow the trees to learn low-level feature combinations that are not possible with HOG.

6 Conclusions

A significant problem in object detection and classification is that non-linear methods are too expensive for many tasks, while the classes are not well separated by linear hyperplanes in existing feature spaces. We approach this problem introducing an intermediate mapping step where examples are mapped from a given feature space to one where they are easier to separate using a linear classifier. We learn boosted trees and induce our feature mapping from the decision rules of the boosted classifier. This way the complex non-linear decision boundary of the boosted classifier can be approximated by a linear one in the extended space. The greedy nature of boosting keeps the mapping sparse.

Using our method, a linear SVM classifier is able to achieve much higher performance than in the original feature space, which confirms the that a wider margin is possible in the new feature space.

Several existing kernel approximation methods are elegant, but we are currently not able to approximate all kernels, and few problems have a known optimal kernel. For a general classification case, we argue that our method is preferred over mapping to kernel-approximating space, since our feature mapping is learnt specifically for the problem, dataset and feature.

In this paper we have focused mainly on exploring and justifying our tree-based mapping algorithms. Since our method can be included in detection and classification frameworks that use linear classifiers, it would be interesting to see how the results translate to detection when used together with a complex linear model such as DPM [5].

References

[1] Omid Aghazadeh, Hossin Azizpour, Josephine Sullivan, and Stefan Carlsson. Mixture component identification and learning for visual recognition. In *Proceedings of the European Conference on Computer Vision*, 2012.

[2] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *Proceedings of the Conference on Computer Vision and Pattern Recognition*, 2005.

[3] Santosh K. Divvala, Alexei A. Efros, and Martial Hebert. How important are 'deformable parts' in the deformable parts model? In *European Conference on Computer Vision (ECCV) 2012, Parts and Attributes Workshop*, 2012.

[4] R.E. Fan, K.W. Chang, C.J. Hsieh, X.R. Wang, and C.J. Lin. LIBLINEAR: A library for large linear classification. *Journal of Machine Learning Research*, 9, 2008.

[5] P. Felzenszwalb, R. Girshick, D. McAllester, and D. Ramanan. Object detection with discriminatively trained part-based models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(9), September 2010.

[6] J. Friedman and B. Popescu. Predictive learning via rule ensembles. *Annals of Applied Statistics*, 2(3), 2008.

[7] J.H. Friedman, Trevor Hastie, and Robert Tibshirani. Additive logistic regression: a statistical view of boosting (With discussion and a rejoinder by the authors). *The Annals of Statistics*, 28, 2000.

[8] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The Elements of Statistical Learning*. Springer, 2011.

[9] S. Lazebnik, C. Schmid, and J. Ponce. Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In *Proceedings of the Conference on Computer Vision and Pattern Recognition*, 2006.

[10] Subhrasnu Maji. Linearized smooth additive classifiers. In *eccv, Workshop on Web-scale Vision and Social Media*, 2012.

[11] Subhrasnu Maji and Alexander C. Berg. Max-margin additive classifiers for detection. In *Proceedings of the International Conference on Computer Vision*, 2010.

[12] Subhrasnu Maji, Alexander C. Berg, and Jitendra Malik. Classification using intersection kernel support vector machines is efficient. In *Proceedings of the Conference on Computer Vision and Pattern Recognition*, 2008.

[13] Tomasz Malisiewicz, Abhinav Gupta, and Alexei A. Efros. Ensemble of exemplar-svms for object detection and beyond. In *Proceedings of the International Conference on Computer Vision*, 2011.

[14] T. Ojala, M. Pietikaainen, and T. Maaenpaaaa. Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 7, 2002.

[15] A Torralba and Alexei A Efros. Unbiased look at dataset bias. In *Proceedings of the Conference on Computer Vision and Pattern Recognition*, 2011.

[16] A. Vedaldi and A. Zisserman. Efficient additive kernels via explicit feature maps. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(3), 2011.

[17] A. Vedaldi and A. Zisserman. Sparse kernel approximations for efficient classification and detection. In *Proceedings of the Conference on Computer Vision and Pattern Recognition*, 2012.

[18] A. Vedaldi, V. Gulshan, M. Varma, and A. Zisserman. Multiple kernels for object detection. In *Proceedings of the International Conference on Computer Vision*, 2009.

[19] Celine Vens and Fabrizio Costa. Random forest based feature induction. In *IEEE International Conference on Data Mining*, 2011.

[20] Stefan Walk, Nikodem Majer, Konrad Schindler, and Bernt Schiele. New features and insights for pedestrian detection. In *Proceedings of the Conference on Computer Vision and Pattern Recognition*, 2010.

[21] Xiaoyu Wang, Tony X. Han, and Shuicheng Yan. An hog-lbp human detector with partial occlusion handling. In *Proceedings of the International Conference on Computer Vision*, 2009.

[22] Liu Yang. Distance metric learning: A comprehensive survey. In *Technical report*, 2006.