Part-Based Feature Synthesis for Human Detection

Aharon Bar-Hillel[1*], Dan Levi[1*], Eyal Krupka[2], and Chen Goldberg[3]

[1] General Motors Advanced Technical Center, Hamanofim 11, Herzliya, Israel
aharon.barhillel,dan.levi@gm.com
[2] Israel Innovation Labs, Microsoft Israel R&D Center, Haifa
eyalk@microsoft.com
[3] Tel Aviv University
chen.goldberg@cs.tau.ac.il

Abstract. We introduce a new approach for learning part-based object detection through feature synthesis. Our method consists of an iterative process of feature generation and pruning. A feature generation procedure is presented in which basic part-based features are developed into a feature hierarchy using operators for part localization, part refining and part combination. Feature pruning is done using a new feature selection algorithm for linear SVM, termed Predictive Feature Selection (PFS), which is governed by weight prediction. The algorithm makes it possible to choose from $O(10^6)$ features in an efficient but accurate manner. We analyze the validity and behavior of PFS and empirically demonstrate its speed and accuracy advantages over relevant competitors. We present an empirical evaluation of our method on three human detection datasets including the current de-facto benchmarks (the INRIA and Caltech pedestrian datasets) and a new challenging dataset of children images in difficult poses. The evaluation suggests that our approach is on a par with the best current methods and advances the state-of-the-art on the Caltech pedestrian training dataset.

Key words: Human detection, Feature selection, Part-Based Object Recognition

1 Introduction

Human detection is an important instance of the object detection problem, and has attracted a great deal of research effort in the last few years [1–6]. It has important applications in several domains including automotive safety and smart video surveillance systems. From a purely scientific point of view, it incorporates most of the difficulties characterizing object detection in general- namely viewpoint, scale and articulation problems. Several approaches have been put forward, with established benchmarks [1, 7] enabling competitive research.

It is widely acknowledged that a method's detection performance largely depends on the richness and quality of the features used, and the ability to combine diverse feature families [3, 8]. While some progress has been made by careful

[*] Both authors contributed equally to this work.

manual feature design [9, 1, 10], there is a growing tendency to automate the feature design and cue integration process. This can be done by feature selection from a very large feature family [2, 11], or by kernel integration methods [8]. The idea of feature selection has been extended to 'feature mining' by introducing the notion of a dynamic hypothesis family to select from, see [3]. In this paper we take this notion one step further.

At an abstract level, we regard automatic feature synthesis as an iterative interplay of two modules: a 'hypothesis generator' and a 'hypothesis selector'. The 'hypotheses generator' gets a temporary classifier and a hypothesis family, and produces an extended hypothesis family, with new features which are conjectured to be helpful to the current classifier. The 'hypotheses selector' then prunes the new suggested feature set and learns a (hopefully better) classifier from it. The distinction between these two agents and the division of labor between them follows similar frameworks in the methodology of scientific discovery ('context of discovery' and 'context of justification' presented in [12]) or in certain forms of reinforcement learning (the actor-critic framework [13]).

This paper makes two main contributions, corresponding to the domains of 'feature generation' and 'feature selection' mentioned above. First, we suggest a part-based feature generation process, where parts are derived from natural images fragments such as those used in [14, 15]. This process starts with basic global features and gradually moves toward more complicated features including localization information, part description refinement, and spatial/logical relations between part detections. More complex part-based features are generated sequentially, from parts proved to be successful in earlier stages. Second, we introduce a new feature selection method for Support Vector Machines (SVMs), termed the SVM Predictive Feature Selection (SVM-PFS), and use it in the pruning stages of the feature synthesis process. SVM-PFS iterates between SVM training and feature selection and provides accurate selection with orders of magnitude speedup over previous SVM wrappers. We provide a formal analysis of SVM-PFS and empirically demonstrate its advantages in a human detection task over alternatives such as SVM-RFE[16], boosting[17] or column generation[18].

We test our feature synthesis process on three human detection datasets, including the two current de-facto benchmarks for pedestrian detection (the INRIA [1] and Caltech [7] datasets) and a difficult dataset of children involved in various activities which we have collected ourselves. Our method is comparable to the best current methods on the INRIA and Children datasets. On the Caltech pedestrian training dataset we achieve a detection rate of 30% at 1 false alarm per image compared to at most 25% for competing methods.

1.1 Overview and related work

The learning process we term *Feature Synthesis* gets positive and negative image window examples as input and learns an image window classifier. *Feature Synthesis* is an iterative procedure in which iteration n is composed of two stages: *feature generation* resulting in a set of candidate features F_n, and *feature selection* resulting in a subset of selected features $S_n \subset F_n$ and a learned linear classifier C_n. In the *feature generation* stage a new set of features T_n is generated, and added to previously generated features. We experimented with two ways to construct the candidate set: the 'monotonic' way $F_n = F_{n-1} \cup T_n$ and the 'non-monotonic' version, in which we continue from the set of previously selected features: $F_n = S_{n-1} \cup T_n$. In the *feature selection* stage the PFS algorithm selects a subset of the candidate features $S_n \subset F_n$ with a fixed size M and returns the learned classifier C_n. From the second iteration on PFS is initialized with the previous selected features and their weights (S_{n-1}, C_{n-1}) directing its search for new useful features. The final classifier consists of the selected features S_n and learned classifier C_n at the final iteration.

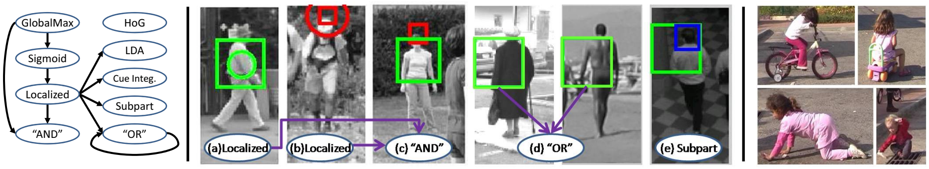

Fig. 1. **Left:** Feature types currently supported in our generation process. An arrow between A and B stands for 'A can be generated from B'. **Center:** Examples of features from our learned classifiers. a,b) Localized features. The rectangle denotes the fragment. The circle marks the 1-std of its location Gaussian. c) Detection example for a spatial "AND" feature composed of fragments a,b. d) Two fragments composing together a semantic "OR" feature. e) A subpart feature. The blue rectangle is the emphasized fragment quarter. **Right:** Typical images from the Children dataset.

While the framework described above can be applied to any feature type, we suggest a process in which the introduced feature sets T_n consist of part-based features with increasing complexity. Part-based representations have attracted a lot of machine vision research [14, 19, 4, 20–22], and are believed to play an important role in human vision [23]. Such representations hold the promise of being relatively amendable to partial occlusion and object articulation, and are among the most successful methods applied to current benchmarks [4, 20]. The parts used in our process are defined using natural image fragments, whose SIFT [9] descriptors are compared to the image in a dense grid [14, 21]. Beginning with simple features corresponding to the global maximal response of a part in the image, we derive more complex features using several operators which can be roughly characterized as based on localization, refinement and combination.

Part localization adds a score to the feature reflecting the location of the part in an absolute framework (commonly referred to as a 'star model' [22]), or with respect to other parts (e.g. in [24]). Part refinement may take several forms: part decomposition into subparts [25], re-training of the part mask for increased discriminative power [4], or binding the SIFT descriptor of the part with additional, hopefully orthogonal, descriptors (e.g. of texture or color). Part combination may take the form of 'and' or 'or' operators applied to component parts, with and without spatial constraints. Applying 'and' operators corresponds to simple monomials introducing non-linearity when no spatial constraints are imposed, and to 'doublets' [24] if such constraints exist. Applying 'or' operators can create 'semantic parts' [21] which may have multiple, different appearances yet a single semantic role, such as 'hand' or 'head'. While most of these feature forms have previously been suggested, here we combine them into a feature generation process enabling successive creation of feature sets with increasing complexity.

The feature synthesis approach proposed requires successive large-scale feature selection and classifier learning epochs. We chose SVM as our base classifier, based on theoretical considerations [26] as well as empirical studies [27]. Feature selection algorithms can be roughly divided into filters, governed by classifier-independent feature ranking, and wrappers, which are selecting features for a specific classifier and include repeated runs of the learner during selection. Typically the former are faster, while the latter are more accurate in terms of classification performance. While several wrapper methods for SVM have been described in the literature [28, 16, 29], all of them require at least one, and usually several SVM runs on the entire data with the full candidate feature set. For large datasets with thousands of examples and features this is prohibitive, as even a single computation of the Gram matrix takes $O(L^2N)$ where L is the number of examples and N is the number of features.

The algorithm we suggest for the *feature selection* stage, SVM-PFS, aims to match the accuracy of existing SVM wrapper methods, and specifically the SVM-RFE [16] method, but with a low computational cost like filter methods. SVM-RFE starts by training SVM with the full candidate feature set, then it removes the features with the lowest absolute weight in a backward elimination process. SVM-PFS uses a similar elimination process, but it avoids training SVM on the entire feature set. Instead SVM is only trained on small subsets of features, and the learned classifier is used to obtain weight predictions for unseen features. Our SVM-PFS analysis derives the predicted weight criterion from gradient considerations, bounds the weight prediction error, and characterizes the algorithm's behavior in the presence of large amounts of useless features. SVM-PFS has a speedup factor of order $C/\log(C)$ over SVM-RFE, where C is the ratio between the sizes of the candidate and final feature sets. In our experiments SVM-PFS accuracy was comparable to SVM-RFE, while speedup factors of up to $X116$ were obtained. This speedup enables our large scale feature synthesis experiments.

There are several lines of work in the literature which are close to our approach in at least one respect. The Deformable Part Model (DPM) [4, 20] shares the part-based nature of the model, and the search for more complex part-based representations. However, the learning technique they employ (latent SVM) is very different from ours, as well as the models learned. Unlike [4, 20], our model typically includes tens to hundreds of parts in the final classifier, with various feature types extracted from them. The 'feature mining' approach [3] shares the attempt to automate hypothesis family formation, and the basic concepts of a dynamic feature set and generator-selector distinction. However, both the generator they employ (parameter perturbations in a single parametric family) and the selector (boosting) are fundamentally different. In the feature selection literature, the column generation approach presented in [18] is the most similar to ours. They use a weight prediction score identical to ours, but in an agglomerative boosting-like framework. Our work is more inspired by RFE both in terms of theory and in adopting an elimination strategy. Though PFS and column generation use the same feature ranking score in intermediate steps, we show that their empirical behavior is very different, with the PFS algorithm demonstrating considerable advantage. Qualitative observations suggest that the reason is the inability of the agglomerative method to remove features selected early, which become redundant later.

We explain the *feature generation* stage in Section 2. Section 3 presents the PFS algorithm for feature selection, Section 4 provides the empirical evaluation of our method and Section 5 briefly discusses future work.

2 Part based feature generation

As a preliminary stage to the first *feature generation stage* we sample a large pool of rectangular image fragments R from the positive training examples, in which the objects are roughly aligned. The fragments cover a wide range of possible object parts with different sizes and aspect ratios as suggested in [14]. Given an image I and a fragment r we compute a sparse set of its detected locations L^r, where each location $l \in L^r$ is an (x, y) image position. The detected locations are computed as in [21]. We compute a 128-dimensional SIFT descriptor [9] of the fragment, $S(r)$ and compare it to the image on a dense grid of locations using the inner product similarity $a^r(l) = S(r) \cdot S(l)$, where $S(l)$ is the SIFT descriptor at location l with the same size as r. From the dense similarity map we compute the sparse detections set L^r as the five top scoring local maxima.

In each *feature generation* stage we generate features of a new type, where each feature is a scalar function over image windows: $f : I \mapsto \mathbb{R}$. The generation function gets the type to generate as input, as well as the previous generated and selected features (F_n,S_n correspondingly) and the fragment pool R, and creates new features of the desired type. For most of the feature types, new features are generated by transforming features from other types, already present in F_n or S_n. The dependency graph in figure 1(Left) shows the generation transformations currently supported in our system. The feature generation order may vary between experiments, as long as it conforms with the dependency graph. In our reported experiments features were generated roughly in the order at which they are presented below, with minor variations. (See Section 4 for more details).

Most of the features we generate represent different aspects of object-part detection, computed using the detection map L^r of one or more fragments. We mark by R_n the set of all the fragments used by the current feature set S_n. Below we describe the feature types implemented and their generation process.

HoG Features. We start with non-part based features obtained by applying HoG descriptors [1] on the entire image window I. The generation of HoG features is independent of the learning state.

GlobalMax Features. Given a fragment r and an image I, $f(I)$ is its maximal appearance score over all the image detections: $f(I) = \max_{l \in L^r} a^r(l)$. One max feature is generated per $r \in R$.

Sigmoid features. We extend each *globalMax* feature by applying a sigmoid function to the appearance score to enhance discriminative power: $f(I) = \max_{l \in L^r} \mathcal{G}(a^r(l))$, where $\mathcal{G}(x) = 1/(1 + \exp(-20 \cdot (x - \theta)))$. Sigmoid function parameter θ was chosen as the *globalMax* feature quantization threshold maximizing its mutual information with the class labels [15].

Localized features. We extend each *sigmoid* feature g by adding a localization score: $f(I) = \max_{l \in L^r} \mathcal{G}(a^r(l)) \cdot \mathcal{N}(l; \mu, \sigma I_{2\times2})$ where $\mathcal{N}$ a 2D Gaussian function of the detection location l. Such features represent location sensitive part detection, attaining a high value when both the appearance score is high and the position is close to the Gaussian mean, similar to parts in a star-like model [22]. Two localized features with $\sigma = 10, 18$ were generated per *sigmoid* feature $g \in F_n$, with μ being the original image location of the fragment r.

Subpart features. A spatial sub-part is characterized by a subset B of the spatial bins in the SIFT descriptor, and specifically we consider the four quarters of the SIFT, with 2×2 spatial bins each. Given a localized feature g we compute the subpart feature as $f(I) = g(I) \cdot S^t(r) \mid_B \cdot S(l_{\max}) \mid_B$ where $l_{\max} \in L^r$ is the argmax location of the maximum operation in $g(I)$. This puts an additional emphasis on the similarity between specific subparts in the detection. We generate four such features for each *localized* feature $g \in S_n$.

LDA features. Given a *localized* feature g, we use the SIFT descriptor $S(l_{\max})$ computed for all training images to train a Linear Discriminant Analysis (LDA)[30] part classifier. The result is a 128 dimensional weight vector w, replacing the original fragment SIFT used in the original localized feature. The LDA feature is hence computed as $f = \max_{l \in L^r} \mathcal{G}(w \cdot S(l)) \cdot \mathcal{N}(l; \mu, \sigma I_{2\times2})$.

"OR" features. For every two *localized* features $g \in S_n$ and $g' \in F_n$ we generate an "OR" feature computed as $f = \max(g, g')$ if their associated fragments originated in similar image locations. Such "OR" features aim to represent semantic object parts with multiple possible appearances. "OR" features with more than two fragments can be created using a recursive "OR" application in which g is already an "OR" feature.

Cue-integration features. Given a localized feature g we compute the co-occurrence descriptor[10] $CO(l_{\max})$ in all training images and train an LDA part classifier using them. The co-occurrence descriptor expresses texture information additional to SIFT, and the feature is computed as an LDA feature, but with $CO(l)$ replacing $S(l)$. Similarly we generate features that integrate both channels by concatenating the SIFT and co-occurrence descriptors.

"AND" features. Given two features based on fragments r, r' we compute their co-detection scores by $f = \max_{l \in L^r, l' \in L^{r'}} a^r(l) \cdot a^{r'}(l') \mathcal{N}_{rel}(l - l') \cdot \mathcal{N}_{abs}((l + l')/2)$. $\mathcal{N}_{rel}, \mathcal{N}_{abs}$ are Gaussian functions preferring a certain spatial relation between the fragments and a certain absolute location, respectively. We generate several hundred such features by choosing pairs in which the score correlation in positive images is higher than the correlation in negative images.

Recall that after each type of features is added, we run the *feature selection* stage which is explained in the next section.

3 Predictive Feature Selection

An SVM gets as input a labeled sample $\{\mathbf{x}_i, y_i\}_{i=1}^{L}$ where $\mathbf{x}_i \in R^M$ are training instances and $y_i \in \{-1, 1\}$ are their labels, and learns a classifier $sign(\mathbf{w} \cdot \mathbf{x} - b)$ by solving the quadratic program

$$\min_{\mathbf{w} \in R^M} \frac{1}{2}||\mathbf{w}||_2^2 + C \sum_{i=1}^{L} \xi_i \quad s.t \ \forall i \ \ y_i(\mathbf{w} \cdot \mathbf{x}_i - b) \geq 1 - \xi_i \tag{1}$$

Denoting the Gram matrix by K (i.e. $K_{ij} = x_i \cdot x_j$), the dual problem is

$$\max_{\alpha \in R^L} ||\alpha||_1 - \frac{1}{2}(\alpha \otimes y)^t K(\alpha \otimes y) \ \ s.t. \ \ 0 \leq \alpha + \eta \leq C \ 0 \leq \eta \,, \ \alpha^t y = 0 \tag{2}$$

where y is the vector of all labels and $\otimes$ stands for the element-wise vector product. Due to Kuhn-Tucker conditions, the optimal weight vector $\mathbf{w}$ can be expressed as $\mathbf{w} = \sum_{i=1}^{L} \alpha_i y_i \mathbf{x}_i$, where α_i are the components of the dual optimum α. Specifically, if we denote by x_i^j the j-th feature of $\mathbf{x}_i$ and by $x^j = (x_1^j, .., x_L^j)$ the entire feature column, then the weight of feature j is given by

$$w_j = \textstyle\sum_{i=1}^{L} \alpha_i y_i x_i^j = \alpha \cdot (y \otimes x^j) \tag{3}$$

Applying this equation to features which were *not* seen during the SVM training can be regarded as weight prediction. The SVM-PFS algorithm (see Algorithm 1) is an iterative pruning procedure which uses the square of the predicted weight as its feature ranking score.

SVM-PFS keeps two feature sets: a working set S with M features, on which SVM is trained, and a candidate feature set F, initially including $N \gg M$ features. In each round, SVM is trained on S, and the resulting α vector is used to compute feature scores. The features in $F \backslash S$ are sorted and the least promising candidates are dropped. Then the working set S is re-chosen by drawing features from the previous S, and from $F \setminus S$, where the ratio between the two parts of the new S is controlled by a stability parameter c. Features from both S and $F \backslash S$ are drawn with a probability proportional to the feature score. The process ends when the candidate feature set size reaches M.

While the algorithm is relatively simple, there are several tricky details worth noting. First, as will be discussed, the weight predictions are 'optimistic', and the actual weights are a lower bound for them. Hence the scores for S, (which are real weights) and for $F \setminus S$ (which are weight predictions) are considered separately in steps 3,4. A second element is the randomness in the choice of S, which is important in order to break the symmetry between nearly identical features and to reduce feature redundancy. The RFE algorithm, which is deterministic, indeed does not cope well with redundant features [31], and PFS does a better job in discarding them. Finally, the L_∞ feature normalization is also important for performance. Beyond making the features comparable, this normalization gives a fixed bound on the radius of the ball containing all examples, which is important for SVM generalization [32]. While the L_2 norm could also be used,

Algorithm 1 The SVM-PFS algorithm

```
Input: $L$ labeled instances $\{\mathbf{x}_i, y_i\}_{i=1}^{L}$ where $\mathbf{x}_i \in R^N$, the allowed $M < N$ , a fraction
parameter $t \in (0,1)$, stability parameter $c \in (0,1)$.
Output: A feature subset $S = \{i_1, .., i_M\}$ and an SVM classifier working on $\mathbf{x}|_s$.

Initialization:
Set the set of candidate features $F = \{1, 2, .., N\}$.
Normalize all the feature columns such that $\forall j \;\; Ex^j = 0\,,\; \|x^j\|_\infty = 1$.
Initialize the working set $S$ by drawing $M$ different random features from $F$.
While $|F| > M$ do
    1. Train an SVM on $\{\mathbf{x}_i|_s, y_i\}_{i=1}^{L}$ and keep the dual weight vector $\alpha$.
    2. For each feature $j \in F$ compute its score $R^j = R(x^j) = \left(\sum_{i=1}^{L} \alpha_i y_i x_i^j\right)^2$.
    3. Sort the scores $\{R^j | j \in F \setminus S\}$ in descending order and drop the last features from
       $F \setminus S$: $F = F \setminus \{j \in F \setminus S \,|\, rank(R^j) > (1-t)|F|\,\} \cup S$,
    4. Choose a new working set $S$ by $S = S_1 \cup S_2$, where
        (a) $S_1$ is chosen from $S$ by drawing $cM$ features without replacement according
            to $p(j) = R^j / \sum_{j \in S} R^j$.
        (b) $S_2$ is chosen from $F \setminus S$ by drawing $(1-c)M$ features without replacement
            according to $p(j) = R^j / \sum_{j \in F \setminus S} R^j$.
Return S and the last computed classifier.
```

L_∞ is preferable as it is biased toward dense features, which are more stable, and it was found superior in all our experiments.

The computational complexity of SVM-PFS can be easily estimated and compared to SVM-RFE [16], which uses the same feature score but employs backward elimination. SVM-PFS includes $R \cong log(M/N)/log(1-t)$ rounds, and the computationally intensive steps in each round are the SVM-training and weight prediction. When the number of features is large, the main component in SVM training is the computation of the kernel matrix at a cost of L^2M. Weight prediction can be done in $\xi L|F|$, where ξ is the proportion of support vectors in the SVM solution. Putting these together and accumulating the computations of all rounds, the number of computations we get for SVM-PFS is $Q_{PFS} = L^2MR + \xi L\,(N - (1-t)M)\,/t$. For $N \gg M$ and large L, i.e. $L \gg N/M$, we have $Q_{PFS} \cong L^2MR$. In comparison, SVM-RFE starts by running SVM on the entire set of N features, and hence the number of computations of SVM-RFE is $Q_{RFE} \cong (N - (1-t)M)\left(L^2 + \xi L\right)/t$. For $N \gg M$ and large L, we get $Q_{RFE} \cong L^2N/t$. Hence, for the typical case where $L \gg N/M$, the theoretical speed gain of SVM-PFS is as follows:

$$\frac{Q_{RFE}}{Q_{PFS}} \cong \frac{L^2N}{t \cdot L^2MR} = \frac{N\log(1-t)}{t \cdot M\log(M/N)} > \frac{N}{M\log_2(N/M)} \tag{4}$$

For the case of $L \ll N/M$, a similar analysis shows that PFS has a speed gain of $O(L/\xi)$. We next present a theoretical analysis of the PFS algorithm.

3.1 Theoretical Analysis

Soft SVM minimizes the loss $L(f) = \frac{1}{2}||w||^2 + C\sum_{i=1}^{L}[1 - y_i f(x_i)]_+$[33], for a classifier $f(x; w, b) = w \cdot x - b$ over a fixed feature set. SVM-PFS, like SVM-RFE, tries to minimize the same SVM loss, but for classifiers $f(x; w, b, S) = w \cdot x \mid_S -b$ restricted to a feature subset S. While SVM-RFE uses real SVM weights and backward elimination, PFS extends this to weight predictions and a combination of forward selection (into S) and backward elimination (applied to F). In this section we justify the usage of weight predictions in forward and backward selection, and elaborate on the stability of PFS when working with large sets with many useless features. Proofs are deferred to the appendix.

Forward Selection: For backward elimination, it is shown in [16] that removing the feature with the smallest actual weight is the policy which least maximizes the SVM loss (in the limit of infinitesimally small weights). Here we give a similar result for forward selection with weight prediction rather than actual weight. Let $f(x; w, b, \{1, .., M\})$ be a classifier trained using soft SVM on the feature set $\{x^j\}_{j=1}^{M}$, and $\{x^j\}_{j=M+1}^{N}$ be a set of additional yet unseen features. In forward selection, our task is to choose the feature index $l \in \{M + 1, .., N\}$ whose addition enables maximal reduction of the loss. Formally, we say that feature x^l is 'optimal in the small weight limit' iff

$$\exists \epsilon_0 > 0 \;\; \forall \epsilon < \epsilon_0 \;\; l = \underset{j \in \{M+1,..,N\}}{\operatorname{argmin}} \; \min_{w,b} L(f(x; (w, \epsilon), b, \{1, .., M\} \cup j)) \qquad (5)$$

The following theorem states the conditions under which the feature with the highest predicted weight is optimal:

Theorem 1. *If both the primal and the dual SVM solutions are unique then* $argmax_{j \in \{M+1,..N\}}(\sum_{i=1}^{K} y_i \alpha_i x_i^j)^2$ *is an optimal feature in the sense of eq. 5.*

The theorem is proved by considering the derivative of SVM's value function w.r.t the parameter perturbation caused by adding a new feature. Note that the non-uniqueness of the primal and dual solutions is an exception rather than the rule for the SVM problem [34].

Backward elimination with weight predictions: PFS uses (noisy) forward selection in choosing S, but most of its power is derived from the backward elimination process of F. While the backward elimination process based on real weights was justified in [16], the utility of the same process with weight predictions heavily depends on the accuracy of these predictions. Let $(\mathbf{w}^{old}, \alpha^{old})$ be the (primal, dual) SVM solution for a feature set S with M features and $(\mathbf{w}^{new}, \alpha^{new})$ be the SVM solution for $S \cup \{M + 1\}$. Using Eq. 3 to predict the weight of the new feature relies on the intuition that adding a single feature usually induces only slight changes to α, and hence the real weight $w^{real} = \alpha^{new} \cdot \hat{x}^{M+1}$ is close to the predicted $w^{pred} = \alpha^{old} \cdot \hat{x}^{M+1}$. The following theorem quantifies the accuracy of the prediction, again in the small weight limit.

Theorem 2. *Assume that adding the new feature* $\mathbf{x}^{M+1}$ *did not change the set of support vectors, i.e.* $SV = \{i : \alpha_i^{old} > 0\} = \{i : \alpha_i^{new} > 0\}$. *Denote by* K *the* *signed Gram matrix, i.e.* $K_{i,j} = y_i y_j x_i \cdot \mathbf{x}_j$*, and by* K_{sv} *the sub-matrix of* K *with lines and columns in* SV*. If* K_{sv} *is not singular (which is the case if* $M > |SV|$ *and the data points are in a general position) then*

$$\frac{\lambda_{min}(K_{sv})}{\|u\|^2 + \lambda_{min}(K_{sv})} w^{pred} \leq w^{real} = \frac{w^{pred}}{1+u^t K_{sv}^{-1} u} \leq w^{pred} \tag{6}$$

where $\mathbf{u} = y \otimes \mathbf{x}^{M+1}|_{sv}$*, and* $\lambda_{min}(K_{sv})$ *is the smallest eigenvalue of* K_{sv}*.*

The assumptions of theorem 2 are rather restrictive, but they often hold when a new feature is added to a large working set (i.e. M is on the order of 10^3). In this case the weight of the new feature is small, and so are the changes to the vector α and the support vector set. The theorem states that under these conditions w^{pred} upper bounds the real weight, allowing us to safely drop features with low predicted weight. Furthermore, as features are accumulated $\lambda_{min}(K_{sv})$ rises, improving the left hand bound in Eq. 6 and entailing better weight prediction. For small M, weight prediction can be improved by adding a small constant ridge to the Gram matrix diagonal, thus raising $\lambda_{min}(K_{sv})$. These phenomena are empirically demonstrated in Section 4.1.

Robustness to noise and initial conditions: In PFS a small subset of features is used to predict weights for a very large feature set, often containing mostly 'garbage' features. Hence it may seem that a good initial S set is required, and that large quantities of bad features in S will lead to random feature selection. The following theorem shows that this is not the case:

Theorem 3. *Assume* S *contains* $M \gg L$ *random totally uninformative features, created by choosing* x_i^j *independently from a symmetric distribution with* $Ex = 0$*,* $Ex^2 = 1$*,* $Ex^4 = K$*. Then with probability approaching 1 as* $M \to \infty$ *all the examples are support vectors and we have*

$$\forall i \ \alpha_i = \frac{1}{M}(1 + \sqrt{(K+L)/M}\xi_i)$$
$$R(x) \propto \rho(x,y)(1 + \sqrt{(K+L)/M}\xi_R)$$

where $\rho(w,z) = (1/\sigma(w)\sigma(z)) \cdot E[(w - Ew)(z - Ez)]$ *is the Pearson correlation between the feature* x *and the labels* y*, and the variables* $\xi_i, \xi_R \sim N(0,1)$*.*

Theorem 3 states that if S contains many 'garbage' features, weight predictions tend toward the Pearson correlation, a common feature selection filter [35]. This guarantees robustness w.r.t to initial conditions and large amounts of bad features. While proved only for $M \gg L$, the convergence of R toward the Pearson coefficient is fast and observed empirically in the first PFS rounds.

4 Empirical evaluation

In Section 4.1 we evaluate the PFS algorithm and compare it to several relevant feature selection methods. We then present human detection results for our full system in Section 4.2.

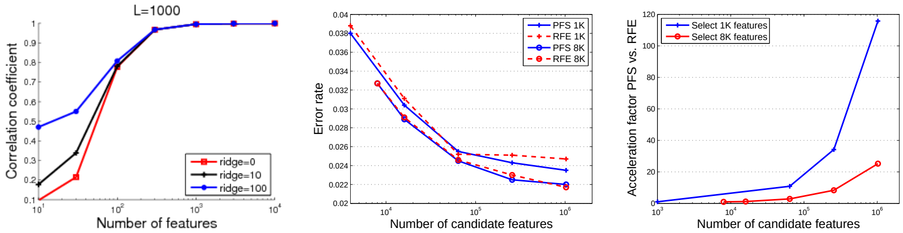

Fig. 2. **Left:** Correlation between predicted and actual weights as a function of working set size M, on an Mnist[36] subset. The features are randomly selected monomials that include two input pixels. Graphs are shown for three ridge (small constant added to the diagonal of the Gram matrix) values. Actual weights are computed by adding the relevant feature to the working set and re-training SVM. **Center:**Accuracy of SVM-RFE and SVM-PFS as a function of the candidate feature set size N for PK-GINA data set. **Right:**Ratio between actual run time of RFE vs. run time of PFS as a function of N for the PK-GINA dataset. PFS runs in less than an hour. The accuracy of both methods is very similar and both benefit from an increasingly larger candidate set. However, the training time of SVM-RFE increases considerably with feature set size, whereas SVM-PFS increases slowly (See Equation 4). PFS shows the greatest gain when selecting from a large candidate set.

4.1 PFS Evaluation

PFS initial evaluation: In all our reported experiments, we use default values for the algorithm parameters of $c = 0.2$ and $t = 0.5$. We used a subset of MNIST, where the task is to discriminate the digits '5' and '8' to check the quality of weight prediction, on which PFS relies. The results, appearing in figure 2(Left) indicated highly reliable weight prediction if the number of features M in S is larger than 500. We then conducted a large scale experiment with the GINA-PK digit recognition dataset from the IJCNN-2007 challenge [37], in which examples are digits of size 28×28 pixels, and the discrimination is between odd and even digits. The features we used are maximum of monomials of certain pixel configurations over a small area. For example, for an area of 2×2 and a monomial of degree 2 the features have the form $\max_{i \in \{0,1\}, j \in \{0,1\}}(P_{x_1+i,y_1+j} P_{x_2+i,y_2+j})$ where $P_{x,y}$ is the pixel value at (x, y) . We generated 1,024,000 features with random parameters. PFS and RFE were evaluated for choosing $M = 1000, 8000$ features out of $N = 8000 - 1,024,000$. The results of these experiments are shown in figure 2(Middle,Right). While the accuracy of PFS and RFE is very similar, PFS was up to $X116$ faster. Notably, our results ranked second best in the challenge (post challenge submission).

Comparison with other feature selection algorithms: We compared PFS to Mutual information max-min[15]+SVM, Adaboost[17], Column generation SVM[18] and SVM-RFE in the task of choosing $m = 500$ features among $N = 19,560$ using a subset of 1700 training samples from the INRIA pedestrian dataset. The feature set included *sigmoid* and *localized* features based on 4000 fragments and *HOG* features. The *per-window* INRIA-test results are presented in figure 3(a). PFS and RFE give the best results, but PFS was an order of magnitude faster. As a further baseline, note that SVM trained over 500 random features achieved a miss rate of 85% at 10^{-4} FPPW, and with 500 features selected using the Pearson correlation filter it achieved 72%.

4.2 Feature synthesis for human detection

Implementation Details. We used 20,000 fragments as our basic fragment pool R, with sizes ranging from 12 × 12 to 80 × 40 pixels. In initial stages we generated a total of $20,000$ *globalMax*, $20,000$ *sigmoid*, $40,000$ *localized* and $3,780$ *HoG*. We then sequentially added the *subparts*, *LDA*, *"OR"*, *cue-integration* and *"AND"* features. In all our experiments we used PFS to choose $M = 500$ features. SVM was used with $C = 0.01$ and a ridge value of 0.01.

INRIA pedestrian [1] results. Following the conclusions of [7] we evaluated our method using both the full-image evaluation based on the PASCAL criteria and the traditional *per-window* evaluation. For the full image evaluation we used a 2-stage classifier cascade to speed up detection. The first stage consists of a HoG classifier [1] adjusted to return 200 detections per image, which are then handed to our classifier. Our non maxima suppression procedure suppresses the less confident window of each pair if their overlap area divided by their union area is greater than $\frac{3}{4}$ or if the less confident window is fully included in the more confident one. We experimented with several mixtures of monotonic and non monotonic steps of the generation process.

The results of our method (**FeatSynth**) are compared to other methods in figure 3(b,c). Further details regarding the tested algorithms and evaluation methodology appear in [6],[7] respectively. These results were obtained using monotonic generation steps in the initial stages (HoG, globalMax, sigmoid and localized features), followed by non monotonic generation steps of *subparts*, *LDA* and *cue-integration* features. At the full image evaluation FeatSynth achieved a 89.3% detection rate at 1 false positive per image (FPPI) outperforming all methods except for LatSvm-V2 [4] (90.7%). Combination features (*"OR"* and *"AND"*) did not add to the performance of this classifier and were therefore omitted. However, these features did contribute when all feature synthesis was done using only non-monotonic steps, which resulted in a slightly worse classifier. Figure 3(d) shows the gradual improvement of the detection rate for the non-monotone process at 1,1/3 and 1/10 FPPI.

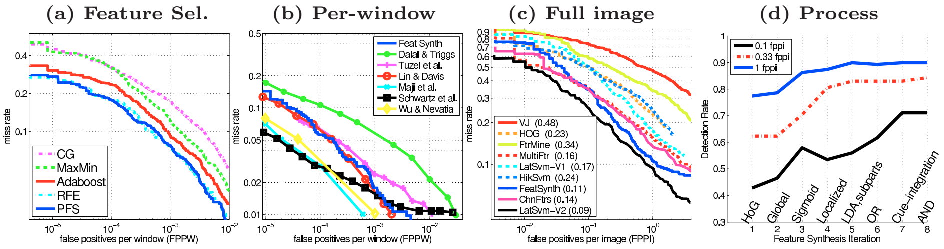
(a) Feature Sel. (b) Per-window (c) Full image (d) Process

Fig. 3. INRIA pedestrian results. (a) Feature selection comparison on a sample of the INRIA dataset. **(b)** Per-window DET curve. **(c)** Full image evaluation on the 288 positive test images. **(d)** Feature synthesis process evaluation on full images. See text for details.

Caltech pedestrian [7] results. The Caltech pedestrian *training* dataset contains six sessions (0-5), each with multiple videos taken from a moving vehicle.

We followed the evaluation methodology from [7], applying our detector on every 30^{th} frame, with a total of 4,306 test images, an order of magnitude larger than the INRIA test. As before we used a 2-stage classifier cascade, with the same classifier trained on the INRIA dataset as the second stage classifier, and the Feature Mining classifier [3], which performs best at low miss rates, as the first stage. Figure 5 shows a comparison to 8 algorithms evaluated in [7] on different subsets of the dataset. We refer the reader to [7] for further details on the tested algorithms and evaluation methodology. In the overall evaluation (fig. 5(a)) FeatSynth achieved a 30% detection rate at 1 FPPI improving the current state-of-the-art, 25%, by MultiFtr [2, 7]. FeatSynth also outperformed the other methods on 50-pixel or taller, un-occluded or partially occluded pedestrians (fig. 5(b)) and on all other dataset subsets except for near pedestrians, proving robustness to varying conditions. Since our classifier (as most other methods) was trained on the INRIA, this evaluation demonstrates its generalization power.

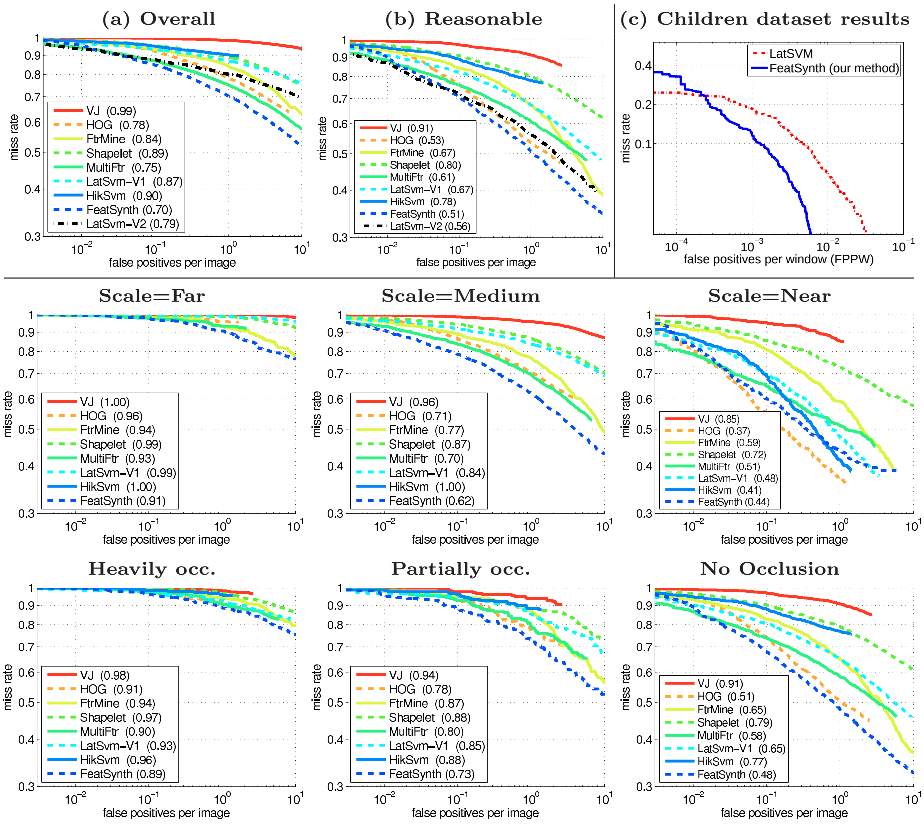

Fig. 4. Results on Caltech pedestrian training dataset and partitions (All except top-right) and child detection results (top-right). See text for details.

Child detection: We created a *Children* dataset of 109 short video clips, of which 82 contain 13 children performing various activities such as crawling, riding a bike or a toy car, sitting and playing. The other 27 clips were taken in the same sites but without the children. These data are mostly relevant for an automotive application of identifying children in the rear camera of a backing vehicle. Based on the videos we compiled a dataset of 2300 children's images, split evenly into train and test, where children from the training set do not appear in the test set. Half the negative images were extracted from the 27 non-children clips, and the other half from the INRIA negative images set. The dataset is rather challenging due to the high pose variance of the children (see figure 1(Left)), making it difficult for the template-based methods used for pedestrians to succeed. The results of our method appear in figure 5(Top-Right), and compared to the part-based method of [4] trained on the children data with 2 components.

5 Conclusions and Future Work

We presented a new methodology for part-based feature learning and showed its utility on known pedestrian detection benchmarks. The paradigm we suggest is highly flexible, allowing for fast exploration of new feature types. We believe that several directions may alleviate the method farther: enable more generation transformations, automate the search over feature synthesis order, introduce negative examples mining into the process, and introduce human guidance as a 'weak learner' into the loop.

References

1. Dalal, N., Triggs, B.: Histograms of oriented gradients for human detection. In: CVPR. (2005)
2. C. Wojec, B.S.: A performance evaluation of single and multi-feature people detection. In: DAGM. (2008)
3. Dollár, P., Tu, Z., Tao, H., Belongie, S.: Feature mining for image classification. In: CVPR. (2007)
4. Felzenszwalb, P., McAllester, D., Ramanan, D.: A discriminatively trained, multiscale, deformable part model. In: CVPR. (2008)
5. Maji, S., Berg, A.C., Malik, J.: Classification using intersection kernel support vector machines is efficient. In: CVPR. (2008)
6. Schwartz, W., Kembhavi, A., Harwood, D., Davis, L.: Human detection using partial least squares analysis. In: ICCV. (2009)
7. Dollár, P., Wojek, C., Schiele, B., Perona, P.: Pedestrian detection: A benchmark. In: CVPR. (2009)
8. Vedaldi, A., Gulshan, V., Varma, M., Zisserman, A.: Multiple kernels for object detection. In: ICCV. (2009)
9. Lowe, D.G.: Distinctive image features from scale-invariant keypoints. IJCV **60** (2004) 91–110
10. Haralick, R., Shanmugam, K., Dinstein, I.: Texture features for image classification. IEEE Transactions on Systems,Man, and Cybernetics **3(6)** (1973)
11. Dollar, P., Tu, Z., Perona, P., Belongie, S.: Integral channel features. In: BMVC. (2009)
12. Popper, K.: Objective knowledge: An Evolutionary Approach. Oxford: Clarendon Press (1972)
13. Kaelbling, L.P., Littman, M.L., Moore, A.W.: Reinforcement learning: A survey. Journal of Artificial Intelligence Research **4** (1996) 237285
14. Ullman, S., Sali, E., Vidal-Naquet, M.: A fragment-based approach to object representation and classification. In: IWVF. (2001)
15. Vidal-Naquet, M., Ullman, S.: Object recognition with informative features and linear classification. (In: ICCV)
16. Guyon, I., Weston, J., Barnhill, S., Vapnik, V.: Gene selection for cancer classification using support vector machines. Machine Learning **46** (2002)
17. Schapire, R.E., Singer, Y.: Improved boosting using confidence-rated predictions. Machine Learning **37** (1999) 297–336
18. Bi, J., Zhang, T., Bennett, K.P.: Column-generation boosting methods for mixture of kernels. In: KDD. (2004)
19. Fergus, R., Perona, P., Zisserman, A.: Object class recognition by unsupervised scale invariant learning. In: CVPR. (2003)
20. Felzenszwalb, P., Girshick, R., McAllester, D., Ramanan, D.: Object detection with discriminatively trained part based models (2010) To appear in the IEEE Transactions on Pattern Analysis and Machine Intelligence.

21. Karlinsky, L., Dinerstein, M., Levi, D., Ullman, S.: Unsupervised classification and part localization by consistency amplification. In: ECCV (2). (2008)
22. Bar-Hillel, A., Weinshall, D.: Efficient learning of relational object class models. IJCV (2008)
23. Tversky, B., Hemenway, K.: Objects, parts, and categories. Journal of Experimental Psychology: General **113(2)** (1984) 169–197
24. Sivic, J., Russell, B.C., Efros, A.A., Zisserman, A., Freeman, W.T.: Discovering objects and their location in images. In: ICCV. Volume 1. (2005)
25. Ullman, S., Epshtein, B.: Visual classification by a hierarchy of extended fragments. In: Toward Category-Level Object Recognition. (2006)
26. Vapnik, V.: The Nature Of Statistical Learning Theory. Springer-Verlag (1995)
27. J.W. Lee, J.B. Lee, M.P., Sonq, S.: An extensive comparison of recent classification tools applied to microarray data. Computational statistics and Data Analysis **48(4)** (2005) 869–885
28. Rakotomamonjy, A.: Variable selection using svm-based criteria. JMLR (2003) 1357–1370
29. Weston, J., Elisseeff, A., Schoelkopf, B., Tipping, M.: Use of the zero norm with linear models and kernel methods. JMLR **3** (2003) 1439–1461
30. Fukunaga, K.: Statistical Pattern Recognition. 2nd edn. Academic Press, San Diego (1990)
31. Z.X Xie, Q.H., Yu, D.: Improved feature selection algorithm based on svm and correlation. In: NIPS. (2006) 1373–1380
32. Shivaswamy, P., Jebara, T.: Ellipsoidal kernel machines. In: AISTATS. (2007)
33. Hastie, T., Tibshirani, R., Friedman, J.: The Elements of Statistical Learning; Data mining, Inference and Prediction. Springer Verlag (2001)
34. Burges, J., Crisp, D.: Uniqueness of the svm solution. In: NIPS. (1999)
35. Hall, M., Smith, L.: Feature subset selection: a correlation based filter approach. In: International Conference on Neural Information Processing and Intelligent Information Systems. (1997) 855–858
36. LeCun, Y., Bottou, L., Bengio, Y., Haffner, P.: Gradient-based learning applied to document recognition. Proceedings of the IEEE **86** (1998) 2278–2324
37. Guyon, I., Saffari, A., Dror, G., Cawley, G.C.: Agnostic learning vs. prior knowledge challenge. In: IJCNN. (2007)

Appendix

Proofs

Theorem 1:

Proof. Consider adding a new feature x^{M+1} with a small *fixed* weight w to the classifier. The minimal loss with the new feature is

$$\begin{aligned} \min_{w_1,..,w_M} \quad & L(\textstyle\sum_{j=1}^M w_j h_j(x) + w x^{M+1}) \\ = \frac{1}{2} w^2 & + \min_{w_1,..,w_M} \tfrac{1}{2} \textstyle\sum_{j=1}^M w_j^2 + C \sum_{i=1}^L \xi_i \end{aligned} \tag{7}$$

$$\begin{aligned} s.t\, \forall i.\ y_i(\textstyle\sum_{j=1}^M w_j x_i^j - b) + \xi_i &\leq 1 - y_i w x_i^{M+1} \\ \xi_i &\geq 0 \end{aligned}$$

This is a perturbed version of the original soft SVM problem over $\boldsymbol{w} = (w_1, ..w_M)$. It has the same optimization argument, and when the constraints are written in the form of $Av \geq B$, the perturbation only occurs in B. Specifically, denote $\hat{x}^{M+1} = y \otimes x^{M+1}$and by X the data matrix of the first M features (i.e. $X_{ij} = x_i^j$). The original SVM problem has

$$A = \begin{pmatrix} -diag(y)X & -I \\ 0 & I \end{pmatrix}$$

$$v = \begin{pmatrix} \boldsymbol{w} \\ \boldsymbol{\xi} \end{pmatrix} \qquad B = \begin{pmatrix} -\mathbf{1} - by \\ 0 \end{pmatrix}$$

and the perturbed version has $B = [-1 - by + w\hat{x}^{M+1}, 0]^t \in R^{2L}$. Consider the problem's value function $F(A, B)$ which maps a matrix×vector (A, B) parameters to the optimal value obtained with these parameters. Theorem 14.2 in [**?**] gives existence conditions and a general formula for the gradient of this function. Specifically it implies that if both the primal and dual solution are unique at (A, B), then F is differentiable at this point and its gradient w.r.t changes in B (for constant A)

is the vector of Lagrange multipliers achieving the dual's optimum. For our problem formulation as $Av \geq B$, this vector is $[-\alpha, \eta]$. Rewriting Eq. 7 using a Taylor approximation we therefore get

$$\begin{aligned} \frac{1}{2}w^2 \quad &+ \quad F(A, B + [w\hat{x}^{M+1}, 0]) \qquad (8) \\ = \tfrac{1}{2}w^2 &+ F(A, B) + [-\alpha, \eta]^t [w\hat{x}^{M+1}, 0] + O(w^2) \end{aligned}$$

For small enough w the $O(w^2)$ terms are negligible and minimizing this expression w.r.t choice of x^{M+1} amounts to minimization of $[-\alpha, \eta]^t[w\hat{x}^{M+1}, 0] = -w\alpha \cdot \hat{x}^{M+1}$. This is equivalent to maximization of $(\alpha \cdot \hat{x}^{M+1})^2$ and choosing $sign(w) = sign(\alpha \cdot \hat{x}^{M+1})$.

Theorem 2:

For simplicity the analysis is described for hard SVM (i.e., without the soft margin term $C\sum_{i=1}^{L} \xi_i$), and without a bias term b in the classifier. However, it can be easily extended to soft SVM. Extending this result to SVM with a bias term is not straightforward using the techniques introduced here. However, they are directly applicable when the bias term is replaced by adding a constant feature column.

Proof. Denote by α_{sv} the reduction of α^{old} to coordinates SV. Then α_{sv} is a maximum of the old dual problem reduced to the variables in SV, i.e. $\alpha_{sv} = \arg\max_{v \geq 0} v \cdot \mathbf{1} - \frac{1}{2}v^t K_{sv} v$. Since $\alpha_{sv} > 0$, it is an interior maximum. The function $v \cdot \mathbf{1} - \frac{1}{2}v^t K_{sv} v$ is concave and has a single interior maximum. Hence α_{sv} is this maximum and can be found by equating the gradient to 0.

$$1 - K_{sv}\alpha_{sv} = 0 \rightarrow \alpha_{sv} = K_{sv}^{-1}\mathbf{1}.$$

The predicted weight for feature $\mathbf{x}^{M+1}$ is $w^{pred} = \alpha^{old} \cdot \hat{x}^{M+1} = \alpha_{sv} \cdot \mathbf{u} = \mathbf{1}^t K_{sv}^{-1}\mathbf{u}$. Under the assumption that SV did not change when feature $\mathbf{x}^{M+1}$ was added, the new α vector is given by $(K_{sv}^{new})^{-1}\mathbf{1}$, where $K^{new} = K + \hat{\mathbf{x}}^{M+1}(\hat{\mathbf{x}}^{M+1})^t$ and $K_{sv}^{new} = K_{sv} + \mathbf{u}\mathbf{u}^t$ is the SV-sub-matrix of K^{new}. The weight of the new feature is hence $w^{real} = \mathbf{1}^t(K_{sv} + uu^t)^{-1}\mathbf{u}$. We get

$$\frac{w^{pred}}{w^{real}} = \frac{\mathbf{1}^t K_{sv}^{-1}\mathbf{u}}{\mathbf{1}^t(K_{sv} + \mathbf{u}\mathbf{u}^t)^{-1}\mathbf{u}} = \frac{\mathbf{1}^t K_{sv}^{-1}\mathbf{u}}{\mathbf{1}^t \left(K_{sv}^{-1} - \frac{K_{sv}^{-1}\mathbf{u}\mathbf{u}^t K_{sv}^{-1}}{1 + \mathbf{u}^t K_{sv}^{-1}\mathbf{u}} \right) \mathbf{u}} = \mathbf{1} + \mathbf{u}^t K_{sv}^{-1}\mathbf{u}$$

where we used *Woodbury's identity* for $(K_{sv} + \mathbf{u}\mathbf{u}^t)^{-1}$. Since K_{sv} is positive semi-definite , $\mathbf{u}^t K_{sv}^{-1}\mathbf{u} \geq \mathbf{0}$, from which the left inequality of the theorem follows. The left inequality follows from the inequality $\mathbf{u}^t K_{sv}^{-1}\mathbf{u} \leq ||\mathbf{u}||^2/\lambda_{min}(K_{sv})$.

Theorem 3:

Proof. The optimal α vector is a solution of a constrained maximization problem:

$$\max_{\alpha \in R^L} \alpha \cdot \mathbf{1} - \frac{1}{2}\alpha^t \hat{K}\alpha \;\; s.t. \;\; 0 \leq \alpha \leq C$$

Where $\hat{K}$ here is the signed Gram Matrix $\hat{K}_{ij} = y_i y_j x_i \cdot x_j$. However, we can solve the maximization problem without the constraints, and if the solution found obeys the constraints then it is clearly also the solution of the constrained problem. The unconstrained solution can be found by plain differentiation to be $\alpha = \hat{K}^{-1}\mathbf{1}$, where the matrix $\hat{K}$ is invertible since $M \gg L$, and the features are sampled from a continuous distribution, hence they are in general position.

We will now analyze the behavior of the Gram matrix entrees. For simplicity of notation, we will drop the hat $\hat{}$ from $\hat{K}$ in the following.

The diagonal entrees: For all $1 \leq i \leq L$ $K_{ii} = \sum_{k=1}^{M}(x_i^k)^2$. As a sum of many independent variables, this quantity has a normal distribution. We have $E[K_{ii}] = \sum_{k=1}^{M} E(x_i^k)^2 = M$. $E[K_{ii}^2]$ is given by

$$E[K_{ii}^2] = E[(\sum_{k=1}^{M}(x_i^k)^2)^2] = E[\sum_{k=1}^{M}\sum_{k'=1}^{M}(x_i^k)^2(x_i^{k'})^2]$$

$$= E[\sum_{k=1}^{M}(x_i^k)^4 + \sum_{k \neq k'}(x_i^k)^2(x_i^{k'})^2] = MK + M(M-1)$$

The variance is hence given by $E[K_{ii}^2] - (E[K_{ii}])^2 = MK + M^2 - M - M^2 = M(K-1)$. This can be summarized by stating that for all i, $K_{ii} \sim N(M, \sqrt{M(K-1)})$.

Off diagonal entrees: Again $K_{ij} = \sum_{k=1}^{M} x_i^k x_j^k$ is normally distributed as a sum of many independent terms. For $i \neq j$ we have $E[K_{ij}] = E[\sum_{k=1}^{M} x_i^k x_j^k] = 0$ since x_i^k, x_j^k are independent for all k. The second moment, which is equal to the variance (as $EK_{ij} = 0$) is given by

$$E[K_{ij}^2] = E[\textstyle\sum_{k=1}^{M} (x_i^k)^2 (x_j^k)^2 + \sum_{k \neq k'} x_i^k x_j^k x_i^{k'} x_j^{k'}] = M$$

Hence $K_{ij} \sim N(0, \sqrt{M})$ for all $i \neq j$.

In addition, for every $1 \leq i, j, l, m \leq L$ we have that K_{ij}, K_{lm} are uncorrelated Gaussians, hence independent. This is trivial for $i \neq l, j \neq m$ due to the independence of the underlying variables, but it also hold if one of these is not true. For example if $j = m$:

$$E[K_{im} K_{lm}] = E[\sum_{k=1}^{M} x_i^k x_m^k \sum_{k'=1}^{M} x_l^{k'} x_m^{k'}]$$

$$= E[\sum_{k=1}^{M} x_i^k x_l^k (x_m^k)^2 + \sum_{k \neq k'} x_i^k x_l^{k'} x_m^{k'} x_m^k] = 0$$

In a similar way we can show the Independence of two matrix elements of the form K_{ii}, K_{ij}.

Based on the above analysis, we can write $K = MI + \sqrt{M}B$, where B is an $L \times L$ matrix with independent elements

$$B(i,i) \sim N(0, \sqrt{K-1})$$
$$B(i,j) \sim N(0,1) \;\; for\, i \neq j$$

Writing $K = M(I + \frac{1}{\sqrt{M}}B)$, and assuming large M, K^{-1} is approximately given by $K^{-1} = \frac{1}{M}(I - \frac{1}{\sqrt{M}}B)$, i.e. this is true up to terms of magnitude $\frac{1}{M}B$ which are negligible for large M. We can now solve for α:

$$\alpha = K^{-1}\mathbf{1} = \tfrac{1}{M}(I - \tfrac{1}{\sqrt{M}}B) \cdot \mathbf{1} = \frac{1}{M}(\mathbf{1} - \frac{\mathbf{1}}{\sqrt{\mathbf{M}}}\mathbf{B} \cdot \mathbf{1}) \tag{9}$$

Consider coordinate i of $B \cdot 1$, $[B \cdot 1]_i = \sum_{i \neq j} B_{ij} + B_{ii}$. This variable is normal since it is a sum of normal variables. We have $E[B \cdot 1_i] = 0$ and from the independence of B's elements we have $var(B \cdot 1_i) = E[(\sum_{i \neq j} B_{ij} + B_{ii})^2] = \sum_{j \neq i} E[B_{ij}^2] + E[B_{ii}^2] = L - 1 + K - 1 = L + K - 2$. Hence $[B \cdot 1]_i \sim N(0, \sqrt{L+K-2})$.

From 9 and these considerations we get for all i that $\alpha_i = \frac{1}{M}(1 - \frac{1}{\sqrt{M}}[B \cdot \mathbf{1}]_i = \frac{1}{M}(1 + \frac{\sqrt{L+K-2}}{\sqrt{M}}\xi_i)$ where we define $\xi_i = -[B \cdot 1]_i / \sqrt{(L+K-2)}$.

Clearly for large enough M, α_i found here obey the constraints $0 \leq \alpha \leq C$, and hence these are the solution of the SVM dual problem.

We now express $R(x)$ using the formula for α. For a feature vector x with $\|x\|_2^2 = 1$ we have:

$$R(x) = \sum_{i=1}^{L} \alpha_i y_i x_i = \tfrac{1}{M}(\textstyle\sum_{i=1}^{L} y_i x_i + \displaystyle\sqrt{\frac{L+K-2}{M}} \sum_{i=1}^{L} y_i x_i \xi_i)$$

Now each term in the last sum is a Gaussian $y_i x_i \xi_i \sim N(0, x_i^2)$ and their sum $\xi_R = \sum_{i=1}^{L} y_i x_i \xi_i$ is therefore a Gaussian with zero mean and variance $\sum_{i=1}^{L} x_i^2 = 1$.

Finally, to see that $\sum_{i=1}^{L} y_i x_i \propto \rho(x,y)$ notice that since $E[x] = \frac{1}{L}\sum_{i=1}^{L} x_i = 0$ and $var(x) = E[x^2] = \frac{1}{L}\sum x_i^2 = \frac{1}{L}$ we have

$$\rho(x,y) = \frac{\frac{1}{L}\sum_{i=1}^{L} y_i x_i - (\frac{1}{L}\sum_{i=1}^{L} x_i) \cdot (\frac{1}{L}\sum_{i=1}^{L} y_i)}{\sigma(x)\sigma(y)} = \frac{\sum_{i=1}^{L} y_i x_i}{\sqrt{L}\sigma(y)}$$

Additional results

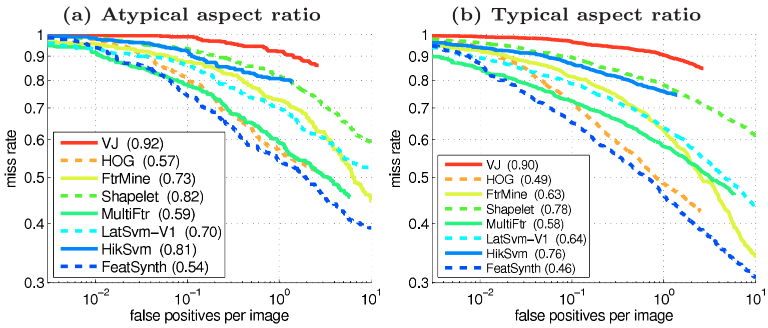

Fig. 5. Additional results on the Caltech pedestrian training dataset partitioned by aspect ratio. Left: Results on the subset of Caltech training dataset including pedestrians with atypical aspect ratios. **Right:** Typical aspect ratios. See [7] for partition and compared method details.