Maximum Margin Linear Classifiers in Unions of Subspaces

Xinrui Lyu[1,2]
xinrui.lyu@epfl.ch
Joaquin Zepeda[1]
joaquin.zepeda@technicolor.com
Patrick Pérez[1]
patrick.perez@technicolor.com

[1] Technicolor
35576, Cesson-Sevigne, France

[2] École Polytechnique Fédérale de Lausanne (EPFL)
CH-1015, Lausanne, Switzerland

Abstract

In this work, we propose a framework, dubbed Union-of-Subspaces SVM (US-SVM), to learn linear classifiers as sparse codes over a learned dictionary. In contrast to discriminative sparse coding with a learned dictionary, it is not the data but the classifiers that are sparsely encoded. Experiments in visual categorization demonstrate that, at training time, the joint learning of the classifiers and of the over-complete dictionary allows the discovery and sharing of mid-level attributes. The resulting classifiers further have a very compact representation in the learned dictionaries, offering substantial performance advantages over standard SVM classifiers for a fixed representation sparsity. This high degree of sparsity of our classifier also provides computational gains, especially in the presence of numerous classes. In addition, the learned atoms can help identify several intra-class modalities.

1 Introduction

The submission of Krizhevsky *et al*. [20] in the 2012 ImageNet Large Scale Visual Recognition Challenge (ILSVRC) has had a tremendous impact on the image classification community and beyond. By obtaining results that outperformed the state-of-the-art at the time [4] by close to ten absolute percentage points, Krizhevsky *et al*. established experimentally that deeply-stacked Convolutional Neural Networks (CNNs) can be used to learn multi-class image classification systems from end to end.

Importantly, almost the entirety of the CNN pipeline can be interpreted as a feature extraction mechanism, with subsequent layers operating on larger patches of the input image in a process reminiscent of local descriptor spatial pyramid pooling methods [16, 24]. It is only the last layer of the CNN which acts as a classifier, and in this respect, recent research efforts have focused less on the classification mechanism in favor of advancing feature extraction methods. The most common classification layer used in CNN architectures is the soft-max classifier [19, 20], but other standard approaches indeed include Support Vector Machine (SVM) based methods such as banks of linear SVMs or ranking SVMs [5, 22, 23]. When using the activation coefficients at the input of the classification layer as a generic feature extractor [14, 21, 32] on new target classes not within the set of training classes, it is indeed

Pages 117.1-117.13
DOI: https://dx.doi.org/10.5244/C.30.117

common to use a standard ℓ_2-penalized linear SVM classifier given its training and testing complexity advantages.

Hence, the importance of the linear SVM in researchers' classification toolbox is not diminished with the arrival of CNN architectures, and hence we propose herein a novel method to learn SVM classifiers. Our approach picks up the line of work on dictionary learning for sparse representations – we propose to learn SVM classifiers that can be represented sparsely with a dictionary which is learned for the classification task. While previous works have addressed learning supervised dictionaries for classification, they have all focused on enforcing the sparsity of representation of the feature vectors and not of the classifiers, like we do.

Forcing the classifier to be sparse in a learned dictionary exposes a number of interesting benefits. One benefit concerns the compactness of the representation – classifiers with compact representations can be stored more efficiently and, importantly, they incur lower computational cost both at training and testing times. Another benefit is that the atoms (columns) of the learned dictionary will inherit semantic properties shared by different classes and hence can often be interpreted as semantic *attributes*, thus opening a possible path to weakly supervised attribute discovery. In a similar manner, atoms of the learned dictionary will often correspond to modalities of the underlying feature distribution that can likewise have interesting semantic interpretations. Forcing the classifier to be sparse using a learned dictionary can also be interpreted as a novel SVM regularization scheme. Unlike other schemes that constrain the norm of the classifier, our regularization requires that all classifiers be represented in terms of a common dictionary, in effect enabling the system to leverage the annotations for all classes when learning any given class.

We evaluate our method using both unsupervised features as well as very recent, CNN-derived features, testing it on well known image classification datasets (PASCAL VOC 2007 [11] and ImageNet [9]). Our experiments establish that our approach results in very sparse representations of classifiers that outperform other SVM classifiers, and with learned dictionaries that carry out automatic attribute and modality discovery as part of the learning process.

The remainder of the paper is organized as follows: In Section 2 we review the literature related to our method, which we then introduce in Section 3 along with a proposed optimization algorithm and benefits. We then present experimental results in Section 4 and concluding remarks in the last section.

Notation: We let $[\mathbf{a}_k]_k$ denote the matrix $[\mathbf{a}_1, \mathbf{a}_2, \ldots]$ and $[a_k]_k$ the row vector $[a_1, a_2, \ldots]$.

2 Background

In this section, we give a brief overview of sparse coding methods and the related supervised and unsupervised dictionary learning algorithms. We also discuss the advantages and drawbacks of various types of SVMs.

Dictionary learning. Dictionaries $\mathbf{D} \in \mathbb{R}^{d \times A}$, with $A > d$, for sparse coding were first learned in an unsupervised manner [1, 10, 26, 30, 36, 40] by approximating the training vectors $\{\mathbf{x}_i \in \mathbb{R}^d\}_{i=1}^M$ with sparse linear combinations of the columns (called *atoms*) of $\mathbf{D}$:

$$\underset{\mathbf{D}}{\operatorname{argmin}} \frac{1}{M} \sum_{i=1}^{M} e(\mathbf{x}_i, \mathbf{D}), \quad \text{where} \quad e(\mathbf{x}, \mathbf{D}) \triangleq \min_{\mathbf{z}} \|\mathbf{x} - \mathbf{D}\mathbf{z}\|_2^2 + \beta \|\mathbf{z}\|_1. \tag{1}$$

Following the *self-taught learning* approach [31], such dictionaries are used to build unsupervised features by aggregating the sparse codes $e(\mathbf{x}, \mathbf{D})$ of image patches into a global

image feature using, for example, max-pooling and spatial pyramids [37].

More recent dictionary learning methods exist that employ a fully supervised framework by relying, for example, on expressions for $\frac{\partial \mathbf{z}}{\partial \mathbf{D}}$. The work of [25] is one of the few to consider a sparse coding objective different from the approximation error $e(\mathbf{x},\mathbf{D})$ in (1), relying instead on a hybrid objective that includes both the classification error and the approximation error. Our method uses rather a purely classification-based sparse-coding objective, dispensing entirely of the reconstruction component of the objective, similar to the objective of ℓ_1-penalized SVMs [42]. Furthermore, our method is the first one to address sparse decompositions of the classifier vectors, as all previous methods have learned dictionaries to decompose the input feature vectors. Our approach can be interpreted as a new SVM regularization scheme, where the regularizer enforces the sparsity of the classifier in a dictionary learned from all training classes. Not only does this offer computational advantages both at training and test times, but it also allows the learning of a few mid-level "attributes" that can be either shared across classes or used to distinguish several modalities within a given class.

Learning linear classifiers. The standard ℓ_2-penalized Support Vector Machine (ℓ_2-SVM) classifier obtained from

$$\underset{(\mathbf{w},b)}{\operatorname{argmin}} \frac{1}{M}\sum_{i=1}^{M} \ell\left(\mathbf{w},b,\mathbf{x}_i,y_i\right) + \frac{\alpha}{2}\|\mathbf{w}\|_2^2, \quad \ell(\mathbf{w},b,\mathbf{x},y) \triangleq \max\left(0,1-y(\mathbf{w}^\top\mathbf{x}+b)\right), \tag{2}$$

where y_i is the label of training vector $\mathbf{x}_i$, can be seen as a sparse linear combination of training vectors (those that are *support vectors*), a consequence of the representer theorem [17]. In practice, however, the number of support vectors is comparable to the dimensionality of the feature space and hence the representation is not truly sparse. Yet the ℓ_2-SVM has one important benefit in that it maximizes the margin between classes, and this translates into better generalization performance.

The ℓ_1-penalized SVM (ℓ_1-SVM), obtained by substituting $\|\mathbf{w}\|_2^2$ with $\|\mathbf{w}\|_1$ in (2), produces weight vectors $\mathbf{w}$ that are sparse. The ℓ_1-SVM has an advantage over the ℓ_2-SVM in classifying high-dimensional feature vectors [38] since the ℓ_1 penalty effectively carries out automatic feature variable selection, accordingly resulting in lower test-time complexity. Also, ℓ_1-SVM outperforms ℓ_2-SVM in the scenarios where the feature vectors are sparse or when there are redundant noise features [42].

The method we present herein can be seen as a way to extend these two benefits of ℓ_1-SVM (test-time complexity and feature selection) to the case where feature vectors are dense, while at the same time retaining the max-margin formulation of ℓ_2-SVM. Similarly to ℓ_2-SVM, our method selects a sparse subset of vectors and forms a linear classifier from a linear combination of this sparse subset. Yet our approach learns an overcomplete dictionary from which to select this subset, further achieving truly sparse representations, whereas the ℓ_2-SVM approach selects this subset from the (overcomplete) set of training vectors, and further requires subsets comparable in size to the dimensionality of the feature space.

Various variants of the SVM problem exist that explore alternative regularization schemes, including the Relative Margin Machine [35], methods relying on Non-negative Matrix Factorization (NMF) [15], the Power SVM method of [41], and various low-complexity solvers relying on hard mining [13] and stochastic gradient descent [34]. It is important to note that the method we propose in this work is complementary to all of the above approaches, and the possible combinations of our method and the above described methods are indeed an interesting research direction.

3 Learning classifiers in unions of subspaces

In this work we propose learning linear classifiers that have the form $\mathbf{w} = \mathbf{D}\mathbf{z}$, where $\mathbf{D} \in \mathbb{R}^{d\times A}$ is the *dictionary* matrix, and $\mathbf{z} \in \mathbb{R}^A$ is a *sparse vector*. We refer to such classifiers as $\mathbf{D}$-*sparse classifiers*. We first focus on learning $\mathbf{D}$-sparse classifiers given $\mathbf{D}$, subsequently proposing a way to learn $\mathbf{D}$.

3.1 Sparse coding of classifiers

Given the dictionary $\mathbf{D}$, learning a linear classifier $\mathbf{D}\mathbf{z}$ that is $\mathbf{D}$-sparse amounts to learning the sparse vector $\mathbf{z}$. A suitable formulation for $\mathbf{z}$ can be derived by substituting $\mathbf{w} = \mathbf{D}\mathbf{z}$ into the original SVM learning objective in (2) and appending an additive sparsity-enforcing penalty term $\beta\|\mathbf{z}\|_1$:

$$\underset{\mathbf{z},b}{\operatorname{argmin}} \frac{1}{M}\sum_{i=1}^{M} f(\mathbf{D},\mathbf{z},b,\mathbf{x}_i,y_i), \text{ where } f(\mathbf{D},\mathbf{z},b,\mathbf{x},y) \triangleq \ell(\mathbf{D}\mathbf{z},b,\mathbf{x},y) + \frac{\alpha}{2}\|\mathbf{D}\mathbf{z}\|_2^2 + \beta\|\mathbf{z}\|_1. \quad (3)$$

The learned linear classifiers $\mathbf{w} = \mathbf{D}\mathbf{z}$ will exist in a union of subspaces, with each subspace being the span of a small subset of atoms from $\mathbf{D}$. Hence we refer to our proposed classifier as a *Union-of-Subspaces SVM* (US-SVM).

Concerning the dimension (equivalent, under mild assumptions, to the number of columns spanning the subspace) of each subspace, we note that it will vary between subspaces.The average dimension over all subspaces, however, will be controlled by the regularization parameter $\beta \geq 0$ which, similarly to $\alpha \geq 0$, needs to be set using cross-validation experiments.

Convexity. A positive consequence of the simplicity of the $\mathbf{D}$-sparse constraint $\mathbf{w} = \mathbf{D}\mathbf{z}$ is that the US-SVM objective in (3) inherits the convexity of the SVM objective in (2): Concerning the first two terms inside the summation of (3), the transformation $\mathbf{w} = \mathbf{D}\mathbf{z}$ is linear in $\mathbf{z}$ and hence $\mathbf{z}$ appears in (3) in the same form as $\mathbf{w}$ appears in (2). Accordingly, these first two terms are convex in $\mathbf{z}$ for losses such as the hinge loss ℓ for which (2) is convex in $\mathbf{w}$. The ℓ_1 penalty in the third term is always convex and, since additions of convex functions are convex, the problem is itself convex.

Special case when $\alpha > 0, \beta = 0$. Two special cases are worth mentioning in relation to (3). The first case is that when $\beta = 0$, which effectively removes the sparsity constraint from the classifier. Assuming that $\mathbf{D}$ is square and full rank, we will have that $\hat{\mathbf{z}} = \mathbf{D}^{-1}\hat{\mathbf{w}}$, where $\hat{\mathbf{w}}$ and $\hat{\mathbf{z}}$ are the optimal solutions, respectively, to (2) and (3).

If $\mathbf{D}$ is instead overcomplete and full rank, there will be an infinite number of solutions $\hat{\mathbf{z}}$ each satisfying the under-determined system: $\hat{\mathbf{w}} = \mathbf{D}\hat{\mathbf{z}}$. Hence, given the convexity of the problem, in the case when $\beta = 0$, one can expect to retrieve a solution $\hat{\mathbf{z}}$ such that $\mathbf{D}\hat{\mathbf{z}}$ is solution of (2) as long as $\mathbf{D} \in \mathbb{R}^{d\times A}$ is full-rank with $A \geq d$.

Special case when $\alpha = 0, \beta > 0$. A second special case is that when α is set to zero. In this situation, we are removing the penalty that forces the ℓ_2 norm of the classifier to be small, and this is the mechanism that ensures that the margin of SVM classifiers is large [29]. Hence this case needs special attention if we are to produce classifiers that generalize well.

In order to gain some insight concerning the margin when $\alpha = 0$, we note that $\|\mathbf{w}\|_2 = \|\mathbf{D}\mathbf{z}\|_2 \leq \|\mathbf{D}\|_2\|\mathbf{z}\|_2$, where $\|\mathbf{D}\|_2$ is the spectral matrix norm. Furthermore, for a positive constant chosen to be $\mu = \max_{\mathbf{x}\neq 0}\frac{\|\mathbf{x}\|_2}{\|\mathbf{x}\|_1}$, we can write $\|\mathbf{x}\|_2 \leq \mu\|\mathbf{x}\|_1, \forall \mathbf{x}$, and hence $\|\mathbf{D}\mathbf{z}\|_2 \leq \mu\|\mathbf{D}\|_2\|\mathbf{z}\|_1$. Defining $\gamma = \frac{2\beta}{\mu\|\mathbf{D}\|_2}$, we have that $\frac{\gamma}{2}\|\mathbf{D}\mathbf{z}\|_2 \leq \beta\|\mathbf{z}\|_1$ and hence, when

setting $\alpha = 0$ in (3), the ℓ_1 penalty term can be interpreted as an upper bound on an implicit penalty $\frac{\gamma}{2}\|\mathbf{Dz}\|_2$, and can thus be expected to have the desired effect on the margin of the classifier. The resulting formulation further enjoys learning complexity advantages related to the removal of the term $\frac{\alpha}{2}\|\mathbf{Dz}\|_2^2$ and related terms in the gradient expressions required to train the classifier (see Section 3.4). We establish this computational advantage empirically in the results section.

3.2 Dictionary learning

The dictionary matrix required in (3) is crucial if one is to obtain classifiers that simultaneously perform well and have a sufficiently sparse representation in $\mathbf{D}$. Hence we now propose a learning algorithm that yields a dictionary $\mathbf{D}$ well suited to the task.

To this end, we assume that we are given a set of training feature vectors $\mathbf{x}_i$ and labels $y_{ik} \in \{-1, 1\}$ indicating the membership (when $y_{ik} = 1$) or absence ($y_{ik} = -1$) of feature vector $i = 1, \ldots, M$ in class $k = 1, \ldots, K$. We also let $\mathbf{y}_i \triangleq [y_{i1}, \ldots, y_{iK}]^\top$, and drop the subscript i when unnecessary.

Given the above training set, we can learn a dictionary by minimizing, over all K classes and all M feature vectors in the training set, the US-SVM objective in (3):

$$\underset{\mathbf{D}\in\mathcal{D},\mathbf{z}_{1:K},b_{1:K}}{\operatorname{argmin}} \frac{1}{KM}\sum_{k=1}^{K}\sum_{i=1}^{M} f\left(\mathbf{D}, \mathbf{z}_k, b_k, \mathbf{x}_i, y_{ik}\right), \tag{4}$$

where $\mathcal{D}$ is the convex set of matrices having columns $(\mathbf{d}_j)_{j=1}^A$ satisfying $\|\mathbf{d}_j\|_2 \leq 1$. Restricting the solution to this set removes scale ambiguity in the choice of $\mathbf{D}$ and $\mathbf{z}$ for a given classifier $\mathbf{Dz}$.

Convexity. Following the same argument as for the problem in (3), one can show that (4) is convex in $\mathbf{z}$ or in $\mathbf{D}$. However, given the fourth-order nature of the $\frac{\alpha}{2}\|\mathbf{Dz}\|_2^2$ penalty term inside f, the problem is not jointly convex in $\mathbf{z}$ and $\mathbf{D}$. Even for the special case where $\alpha = 0$ and this fourth order term disappears, the problem is not convex for common loss functions such as the hinge loss due to a substraction of a second order term involving $\mathbf{Dz}$.

3.3 Elastic net and non-negativity constraint

We further consider two variants of the learning problems presented in (3) and (4). In the first variant, we substitute the ℓ_1 penalty term by an elastic net penalty term. This amounts to substituting f in (3) and (4) with

$$f(\mathbf{D}, \mathbf{z}, b, \mathbf{x}, y) \triangleq \ell(\mathbf{Dz}, b, \mathbf{x}, y) + \frac{\alpha}{2}\|\mathbf{Dz}\|_2^2 + \beta\left(r\|\mathbf{z}\|_1 + (1-r)\|\mathbf{z}\|_2^2\right), \quad r \in [0, 1]. \tag{5}$$

The elastic net approach makes the $\mathbf{z}$-related penalty term strictly convex, thus providing a unique solution of $\mathbf{z}$ when $\mathbf{D}$ is fixed [43].

We further consider restricting the sparse coefficients to be non-negative, $\mathbf{z}, \mathbf{z}_1, \ldots, \mathbf{z}_K \in \mathbb{R}_+^A$. One motivation for this is that it reduces the number of local minima in the dictionary learning problem of (4), as a given solution $\mathbf{Dz}$ can in general only be achieved with a specific polarity of $\mathbf{D}$ when $\mathbf{z}$ is sufficiently sparse.

In the experiments section, we use US-SVM to refer to the original formulation in (3) and (4), and US-SVM-E, US-SVM-N, and US-SVM-NE to refer, respectively, to the variants using the elastic net regularizer, the non-negative constraint on the sparse coefficients, and both of these simultaneously.

3.4 Algorithm

We propose solving both problems (3) and (4) using Stochastic Gradient Descent (SGD) as it is very efficient in situations where training data is abundant [33]. SGD can be used to solve problems of the form $g(\boldsymbol{\theta},\mathcal{S}) = \frac{1}{M}\sum_{i=1}^{M}\Phi(\boldsymbol{\theta},\mathcal{S}_i)$, where $\boldsymbol{\theta}$ denotes the parameters that are being learned, $\mathcal{S}$ the annotated training set and $\mathcal{S}_i$ one of the M training samples (for classification, $\mathcal{S}_i = (\mathbf{x}_i,\mathbf{y}_i)$). At iteration t, SGD draws a random training sample $\mathcal{S}_{i_t}$ and updates the parameters $\boldsymbol{\theta}$ using $\boldsymbol{\theta}^t = \boldsymbol{\theta}^{t-1} - \lambda_t \nabla_{\boldsymbol{\theta}}\Phi(\boldsymbol{\theta}^{t-1},\mathcal{S}_{i_t})$. We use learning rates of the form $\lambda_t = \lambda/(t+t_0)$ [3], finding suitable values for λ and t_0 using cross-validation.

For the case when the number of parameters is too large, or when complicated dependencies between parameters make differentiation difficult, one can use a Block-Coordinate SGD (BC-SGD) variant. Letting $\boldsymbol{\theta} = (\boldsymbol{\theta}_1,\ldots,\boldsymbol{\theta}_Q)$, BC-SGD consists of updating the subset of parameters in only one of the blocks $\boldsymbol{\theta}_{q_t}$ at any given iteration t when applying SGD, alternatively repeating the same block B times before moving to the next block.

Obtaining sparse z with SGD. Bottou [3] briefly discusses an SGD ℓ_1-SVM solver that is very efficient and that we adapt to obtain sparse solutions $\mathbf{z}$. The approach consists of representing $\mathbf{z}$ as a difference of non-negative vectors, $\mathbf{z} = \mathbf{u} - \mathbf{v}$ with $\mathbf{u},\mathbf{v} \in \mathbb{R}^A_+$, accordingly changing the ℓ_1 penalty $\beta\|\mathbf{z}\|_1$ to $\beta(\|\mathbf{v}\|_1 + \|\mathbf{u}\|_1)$. Note that, at the optimum, $\mathbf{u}$ and $\mathbf{v}$ will have disjoint supports and hence $\|\mathbf{z}\|_1 = \|\mathbf{u}\|_1 + \|\mathbf{v}\|_1$. At each SGD step, the corresponding algorithm works by updating $\mathbf{u}$ and $\mathbf{v}$ using the related sub-gradient, followed by a projection onto the set of non-negative vectors.

Since often the resulting sparse vectors will be only nearly sparse, following the learning procedure, we further prune from $\mathbf{z}$ the smaller-energy coefficients by specifying a target sparsity level.

When solving the US-SVM problem in (3) using SGD, we let $\boldsymbol{\theta} = (\mathbf{u},\mathbf{v},b)$ and $\Phi(\boldsymbol{\theta},\mathbf{x},y) = f(\mathbf{D},\mathbf{u}-\mathbf{v},b,\mathbf{x},y)$. When solving (4), $\boldsymbol{\theta} = (\mathbf{D},\mathbf{u}_1,\mathbf{v}_1,b_1\ldots,\mathbf{v}_K,\mathbf{u}_K,b_K)$ and $\Phi(\boldsymbol{\theta},\mathbf{x},\mathbf{y}) = \sum_{k=1}^{K} f(\mathbf{D},\mathbf{u}_k-\mathbf{v}_k,b_k,\mathbf{x},y_k)$.

Gradient expressions. For completeness, we note that, when solving the US-SVM problem in (3) using SGD, $\boldsymbol{\theta} = (\mathbf{u},\mathbf{v},b)$ and we use

$$\Phi_1(\boldsymbol{\theta},\mathbf{x},y) = f(\mathbf{D},\mathbf{u}-\mathbf{v},b,\mathbf{x},y), \tag{6}$$

where we employ the definition of f given in (5) as it is more general. Accordingly, the sub-gradient required for SGD when using the decomposition $\mathbf{z} = \mathbf{u} - \mathbf{v}$ with $\mathbf{u},\mathbf{v} \in \mathbb{R}^A_+$ can be assembled from

$$\begin{aligned}\nabla_{\mathbf{u}}\Phi_1 &= -\hat{y}\mathbf{D}^\top\mathbf{x} + \alpha\mathbf{D}^\top\mathbf{D}\mathbf{z} + r\beta\mathbf{1} + 2(1-r)\mathbf{z},\\ \nabla_{\mathbf{v}}\Phi_1 &= +\hat{y}\mathbf{D}^\top\mathbf{x} - \alpha\mathbf{D}^\top\mathbf{D}\mathbf{z} + r\beta\mathbf{1} - 2(1-r)\mathbf{z},\end{aligned} \tag{7}$$

where $\mathbf{1}$ is the ones vector and

$$\hat{y} = y \text{ if } y\big((\mathbf{D}\mathbf{z})^\top\mathbf{x} + b\big) < 1,\ 0 \text{ otherwise.} \tag{8}$$

We can likewise derive $\hat{\mathbf{y}}$ from $\mathbf{y}$ by applying the above expression in an element-wise manner.

When solving (4), $\boldsymbol{\theta} = (\mathbf{D},\mathbf{u}_1,\mathbf{v}_1,b_1\ldots,\mathbf{v}_K,\mathbf{u}_K,b_K)$ and we use

$$\Phi_2(\boldsymbol{\theta},\mathbf{x},\mathbf{y}) = \sum_{k=1}^{K} f(\mathbf{D},\mathbf{u}_k-\mathbf{v}_k,b_k,\mathbf{x},y_k). \tag{9}$$

Algorithm 1 SGD algorithm for generic US-SVM

Input: Training set $\mathcal{S} = \{(\mathbf{x}_i, \mathbf{y}_i)\}_{i=1\cdots M}$, $\alpha, \beta \in \mathbb{R}$ (regularization parameters), $\lambda_{\mathbf{Z}}$, $\lambda_{\mathbf{D}}$, $t_{\mathbf{Z},0}$, $t_{\mathbf{D},0}$ (learning rate parameters), μ (learning rate multiplier for bias), r (elastic-net parameter), and $\mathbf{D} \in \mathbb{R}^{d\times A}$ (initial dictionary), $\mathbf{U} \in \mathbb{R}_+^{A\times K}, \mathbf{V} \in \mathbb{R}_+^{A\times K}$ (inital sparse code) and $\mathbf{b} \in \mathbb{R}^K$ (initial bias), E (number of epochs), B (number of iterations per block)

$t := 0$
while $t < EM$ **do**
  for 1 **to** B **do**
    Draw $(\mathbf{x}, \mathbf{y})$ from $\mathcal{S}$ randomly and unrepeatedly
    Compute $\hat{\mathbf{y}}$ (Eq.8)
    $\mathbf{Z} = \mathbf{U} - \mathbf{V}$
    Update the sparse codes:
    $\mathbf{U} \leftarrow [\mathbf{U} - \lambda_{\mathbf{Z},t}(r\beta\mathbf{J} - \mathbf{D}^\top\mathbf{x}\hat{\mathbf{y}}^\top + \alpha\mathbf{D}^\top\mathbf{D}\mathbf{Z} + 2(1-r)\mathbf{Z})]_+$
    $\mathbf{V} \leftarrow [\mathbf{V} - \lambda_{\mathbf{Z},t}(r\beta\mathbf{J} + \mathbf{D}^\top\mathbf{x}\hat{\mathbf{y}}^\top - \alpha\mathbf{D}^\top\mathbf{D}\mathbf{Z} - 2(1-r)\mathbf{Z})]_+$
    Update the bias: $\mathbf{b} \leftarrow \mathbf{b} + \mu\lambda_{\mathbf{Z},t}\hat{\mathbf{y}}$
    $t \leftarrow t + 1$
  end for
  for 1 **to** B **do**
    Draw $(\mathbf{x}, \mathbf{y})$ from $\mathcal{S}$ randomly and unrepeatedly
    Compute $\hat{\mathbf{y}}$ (Eq.8)
    $\mathbf{Z} = \mathbf{U} - \mathbf{V}$
    Update the dictionary:
    $\mathbf{D} \leftarrow \mathbf{D} - \lambda_{\mathbf{D},t}(-\mathbf{x}\hat{\mathbf{y}}^\top\mathbf{Z}^\top + \alpha\mathbf{D}\mathbf{Z}\mathbf{Z}^\top)$
    Project $\mathbf{D}$ to $\mathcal{D}$:
    for $j = 1$ **to** A **do**
      $\mathbf{d}_j \leftarrow \mathbf{d}_j / \max(\|\mathbf{d}_j\|, 1)$
    end for
    $t \leftarrow t + 1$
  end for
end while

The gradient required for SGD can be assembled from the partial gradients below, where we use Φ_1 from (6):

$$\nabla_{\mathbf{D}}\Phi_2 = -\sum_{k=1}^{K}\left(\hat{y}_k\mathbf{x}\mathbf{z}_k^\top + \alpha\mathbf{D}(\mathbf{z}_k\mathbf{z}_k^\top)\right), \quad \nabla_{\mathbf{u}_k}\Phi_2 = \nabla_{\mathbf{u}}\Phi_1|_{\mathbf{u}=\mathbf{u}_k}, \; \nabla_{\mathbf{v}_k}\Phi_2 = \nabla_{\mathbf{v}}\Phi_1|_{\mathbf{v}=\mathbf{v}_k}. \quad (10)$$

We note that other choices for $\Phi(\boldsymbol{\theta}, \mathcal{S}_i)$ are possible for this second problem, and that using BC-SGD can reduce the complexity of the learning process. A common approach in dictionary learning, for example, consists of updating a single atom at a time [1].

In Algorithm 1, $\mathbf{J} = [1]_{ij}$ is an all-one matrix of size $A \times K$ and we summarize an algorithm that solves (4) for the general case of elastic net penalization (US-SVM-E); Note that a solution for (3) follows by setting the learning rate $\lambda_{\mathbf{D}}$ to 0. The same algorithm can be used to implement the non-negativity of $\mathbf{z}$ by forcing $\mathbf{V} = \mathbf{0}$ and not updating it. Hence the same algorithm addresses all four US-SVM variants (US-SVM, US-SVM-E, US-SVM-N, US-SVM-NE).

3.5 Benefits of the proposed method

Large-scale image retrieval has been made possible by developments in approximate nearest neighbor search relying on compact (vector-quantized) representations of the image feature vectors [18]. These approaches have been applied to enable large-scale search using image classifiers [7] by likewise representing the feature database compactly while using a standard dense linear classifier. The approach we present herein is complementary in that it enables using linear classifiers that are sparse. Letting $\mathbf{X} = [\mathbf{x}_i]_i$ represent the feature database, the resulting test-time operation $\mathbf{w}^\top\mathbf{X} = (\mathbf{D}\mathbf{z})^\top\mathbf{X} = \mathbf{z}^\top(\mathbf{D}^\top\mathbf{X})$ can be carried out very efficiently

by pre-computing $\mathbf{X}' \triangleq \mathbf{D}^\top \mathbf{X}$, leaving only the low-complexity sparse-vector/matrix product $\mathbf{z}^\top \mathbf{X}'$ for test time. If one further needs to store a bank of these classifiers, a sparse representation such as $\mathbf{z}$ is beneficial in that it has small storage footprint.

Our US-SVM formulation further enjoys reduced learning complexity over ℓ_2-SVM in situations involving large, fixed negative sets [2, 13, 27, 28, 39]. In these cases, $\mathbf{x}'_i \triangleq \mathbf{D}^\top \mathbf{x}_i$ for all negative vectors $\mathbf{x}_i$ can be computed once for all learning runs, and one need only carry out lower-complexity operations of the form $\mathbf{z}^\top \mathbf{x}'_i$ during the learning process.

Another interesting property of US-SVM is that, since the atoms of dictionaries learned from (4) are shared by multiple classes, it is possible to interpret these as attribute classifiers. We have also observed that, for classes exhibiting multi-modal feature distributions, our method can discover these modalities. We provide empirical evidence of both of these properties in the results section.

4 Experiments

In this section, we first discuss how we choose the hyperparameters for our model, and then compare the performance of ℓ_1-SVM and ℓ_2-SVM on different datasets with that of US-SVM, US-SVM-N, US-SVM-E, and US-SVM-NE. The performance is evaluated in terms of mean Average Precision (mAP) [12] and sparsity (number of nonzero coefficients in the representation). To ensure that insignificant coefficients do not adversely influence sparsity, for all classifiers, we vary the sparsity level by pruning the lower-energy coefficients.

We present results using two datasets: PASCAL VOC 2007 (PVOC) and ImageNet. From ImageNet, we derive two datasets: one consists of 200 synsets randomly chosen from ImageNet (ImageNet-200) and the other consists of 20 synsets (ImageNet-20) randomly chosen but without overlap with ImageNet-200. We represent images using VGG-128 features [5] and VLAD features [8]. The VLAD features are of very large dimension (8 192), and hence we reduce their dimension to 128 by means of PCA.

Hyperparameter cross-validation. Our proposed model has 8 hyperparameters: α (ℓ_2 regularization parameter), β (ℓ_1 regularization parameter), $\lambda_{\mathbf{z}}$, $\lambda_{\mathbf{D}}$, $t_{\mathbf{z},0}$, $t_{\mathbf{D},0}$ (parametrization coefficients for learning rates of $\mathbf{z}$ and $\mathbf{D}$), r (elastic-net parameter) and A (size of the dictionary). Two of these hyperparameters (the $t_{\bullet,0}$s) are set empirically based on training cost over a small subset of the training samples. By means of cross-validation, we further found that only three of the remaining hyperparameters (β, $\lambda_{\mathbf{D}}$ and $\lambda_{\mathbf{z}}$) are most important and should be set by cross-validation. Concerning α and the number of atoms A, a good empirical strategy is to set the number of atoms to a multiple of the feature dimension and α to a value close to zero (see Table 1 and related discussion).

Evaluation on PASCAL VOC. In Fig. 1 (*left*), we evaluate our proposed US-SVM learning method. We learn both $\mathbf{D}$ and the $\mathbf{z}_i$ by solving (4) over the PASCAL VOC dataset, using VLAD features. The aim of US-SVM is to provide improved performance at low search complexities (or accordingly, low classifier sparsity values). The experiments illustrate that US-SVM indeed offers a performance advantage for lower sparsity values. For higher sparsity levels, ℓ_2-SVM enjoys a performance advantage as is to be expected from the dense nature of VLAD features. Yet using dense classifiers becomes too expensive when searching in very large image sets.

Evaluation on ImageNet. In Fig. 1 (*right*) we evaluate how well dictionaries learned on an auxiliary training set (ImageNet-200) by means of (4) transfer to new target datasets (ImageNet-20), where only the sparse representation is learned using (3). This use case

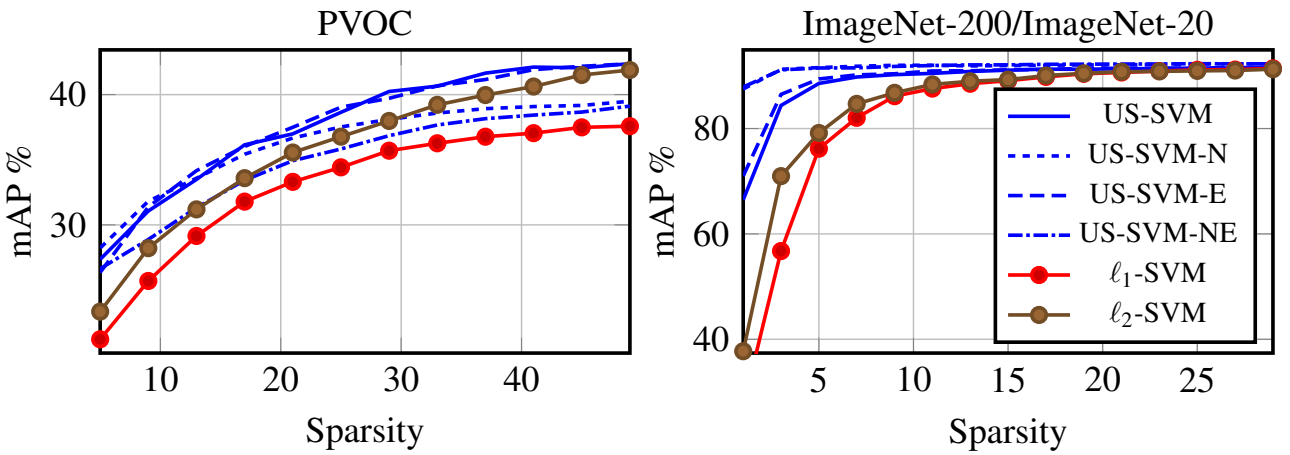

Figure 1: Classification performance of different types of SVMs versus classifiers sparsity (average $|\mathbf{z}_k|_0$). *Left*: VLAD features, $A = 100$, PVOC. *Right*: VGG-128 features, $A = 300$, $\mathbf{D}$ learned on ImageNet-200, results presented for classifiers $\mathbf{z}_k$ learned and tested on ImageNet-20.

Figure 2: Examples of visual attributes captured by learned atoms across multiple classes. *Top*: water; *middle*: rectangular shape; *bottom*: round shape. Each row presents the ten images with highest $\mathbf{d}_{a_j}^\top \mathbf{x}$ score for three different atoms $\mathbf{d}_{a_j}, j = 1,2,3$. For each row, images with the same border color belong to the same class.

corresponds to real search scenarios where the dictionary matrix $\mathbf{D}$ is learned offline from a large auxiliary dataset. At search time, only the sparse code $\mathbf{z}$ for a never-before-seen class needs to be learned. The resulting sparse code can be applied efficiently to a large set of search feature vectors $\mathbf{X} = [\mathbf{x}_i]_i$ using $\mathbf{z}^\top \mathbf{D}^\top \mathbf{X} = \mathbf{z}^\top \mathbf{X}'$, where $\mathbf{X}' \triangleq \mathbf{D}^\top \mathbf{X}$ is precomputed using a transformation $\mathbf{D}$ that enables the employment of very sparse $\mathbf{z}$. Note that, indeed, the US-SVM classifiers in Fig. 1 (*right*) enjoy nearly constant performance for drastically low sparsity levels of less than 5 (*i.e.*, lower than 4%), where the performance is close to 20 mAP points better (a +20% difference) than that of ℓ_1-SVM or ℓ_2-SVM.

Attribute discovery. Fig. 2 shows examples of visual "attributes" (associated to atoms) that are learned automatically using our proposed method. Each row of images corresponds to the top-10 ranked images when using the corresponding atom alone as a classifier: Letting $\mathbf{x}$ represent a generic image feature, the images are ranked based on the $\mathbf{d}_{a_j}^\top \mathbf{x}$ score for three different atoms $\mathbf{d}_{a_j}, j = 1,2,3$ (one per row) chosen for visualization purposes. Note that the classes corresponding to these top-ranked are very varied, indicating that the atom has not specialized to a given class. Yet common visual elements are evident for the top-ranked images of each row, whether geometric (round or rectangular parts) or texture-based (water).

Modality discovery. In Fig. 3 we illustrate how our method can automatically discover intra-class modalities by ranking images of class *ping-pong ball* according the the highest-

Figure 3: Different modalities discovered by learned atoms on the class "ping-pong ball" from ImageNet. *Red*: close-ups of ping-pong balls; *green*: casual ping-pong matches indoor; *blue*: formal ping-pong matches in a gymnasium. Each groups of five images presents the five images from class *ping-pong ball* with highest $\mathbf{d}_{a_j}\mathbf{x}$ score for the three atoms $\mathbf{d}_{a_j}, j = 1,2,3$ with highest $|z_{a_j}|$ from the US-SVM representation $\mathbf{Dz} = [\mathbf{d}_a]_a[z_a]_a^\top$ of class *ping-pong ball*.

	$A = 100$		$A = 200$		$A = 300$	
mAP (%)	40.49	42.00	42.86	43.40	43.50	43.63
Runtime (sec/epoch)	2.62	4.63	3.13	6.02	3.69	20.74

Table 1: Runtime performance and mAP for $A = 100, 200$ and 300 when *(left column)* $\alpha = 0$ and when *(right column)* cross-validating α.

energy atoms in the corresponding US-SVM classifier. Each row presents the five images from class *ping-pong ball* with highest $\mathbf{d}_{a_j}^\top\mathbf{x}$ score for the three atoms $\mathbf{d}_{a_j}, j = 1,2,3$ with highest $|z_{a_j}|$ from the US-SVM representation $\mathbf{Dz} = [\mathbf{d}_a]_a[z_a]_a^\top$ of class *ping-pong ball*. Note that sub-modalities of the class are evident from the top-ranked images from each atom.

Runtime improvement when $\alpha = 0$. In Table 1, we evaluate run-time improvements when comparing US-SVM to the $\alpha = 0, \beta > 0$ variant dubbed US-SVM-ℓ_1. This special case enjoys the added advantage of reduced learning complexity and is hence important in situations such as on-the-fly search [6] where the user needs to wait for classifiers to train. The table establishes that indeed, setting $\alpha = 0$ can result in important runtime advantages without significant sacrifice of searching performance, as is to be expected since the special case when $\alpha = 0, \beta > 0$ amounts to maximization of a margin upper bound (see Section 3.1).

5 Conclusion

In this work, we introduced the Union-of-Supspaces Support Vector Machine (US-SVM), an approach that embeds supervised dictionary learning into an SVM learning objective. Contrary to existing approaches in supervised dictionary learning, our learned dictionary does not encode the data vectors, but rather the classifiers. We introduce several variants of the proposed algorithm, and apply our method to the task of visual categorization using standard datasets, establishing experimentally that our approach can provide substantial improvements in performance at low representation sparsities. We further show empirically that the learned dictionaries implicitly perform automatic discovery of attributes that are shared across classes, as well as automatic discovery of modalities in the data vector distributions.

References

[1] Michal Aharon, Michael Elad, and Alfred Bruckstein. K-SVD: An algorithm for designing overcomplete dictionaries for sparse representation. *IEEE Transactions on Signal Processing*, 54(11):4311–4322, 2006.

[2] Yusuf Aytar and Andrew Zisserman. Enhancing exemplar SVMs using part level transfer regularization. In *British Machine Vision Conference*, 2012.

[3] Leon Bottou. Stochastic gradient descent tricks. In *Neural networks: Tricks of the trade*, volume 1. Springer, 2nd edition, 2012.

[4] Ken Chatfield, Victor Lempitsky, Andrea Vedaldi, and Andrew Zisserman. The devil is in the details: An evaluation of recent feature encoding methods. In *British Machine Vision Conference*, 2011.

[5] Ken Chatfield, Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Return of the devil in the details: Delving deep into convolutional nets. In *British Machine Vision Conference*, 2014.

[6] Ken Chatfield, Relja Arandjelović, Omkar Parkhi, and Andrew Zisserman. On-the-fly learning for visual search of large-scale image and video datasets. *International Journal of Multimedia Information Retrieval*, 4(2):75–93, 2015.

[7] Ken Chatfield, Karen Simonyan, and Andrew Zisserman. Efficient on-the-fly category retrieval using convnets and GPUs. In *Asian Conference on Computer Vision*, 2015.

[8] Jonathan Delhumeau, Philippe-Henri Gosselin, Hervé Jégou, and Patrick Pérez. Revisiting the VLAD image representation. In *ACM Multimedia*, 2013.

[9] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. ImageNet: A large-scale hierarchical image database. In *Computer Vision and Pattern Recognition*, 2009.

[10] Kjersti Engan, Sven Ole Aase, and J Hakon Husoy. Method of optimal directions for frame design. In *International Conference on Acoustics, Speech, and Signal Processing*, 1999.

[11] Mark Everingham, Luc Van Gool, Christopher Williams, John Winn, and Andrew Zisserman. The Pascal visual object classes (VOC) challenge. *International Journal of Computer Vision*, 88(2):303–338, 2010.

[12] Mark Everingham, S. M. Ali Eslami, Luc Van Gool, Christopher Williams, John Winn, and Andrew Zisserman. The Pascal visual object classes challenge: A retrospective. *International Journal of Computer Vision*, 111(1):98–136, 2014.

[13] Pedro Felzenszwalb, Ross Girshick, David McAllester, and Deva Ramanan. Object detection with discriminatively trained part-based models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(9):1627–1645, 2010.

[14] Yunchao Gong, Liwei Wang, Ruiqi Guo, and Svetlana Lazebnik. Multi-scale orderless pooling of deep convolutional activation features. In *European Conference on Computer Vision*, 2014.

[15] Mithun Das Gupta, Jing Xiao, and San Jose. Non-negative matrix factorization as a feature selection tool for maximum margin classifiers. In *Computer Vision and Pattern Recognition*, 2011.

[16] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Spatial pyramid pooling in deep convolutional networks for visual recognition. In *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 37(9):19014–1916, 2015.

[17] Thomas Hofmann, Bernhard Scholkopf, and Alexander J. Smola. Kernel methods in machine learning. *The Annals of Statistics*, 36(3):1171–1220, 2008.

[18] Hervé Jégou, Matthijs Douze, Cordelia Schmid, and Patrick Pérez. Aggregating local descriptors into a compact image representation. In *Computer Vision and Pattern Recognition*, 2010.

[19] Yangqing Jia, Evan Shelhamer, Jeff Donahue, Sergey Karayev, Jonathan Long, Ross Girshick, Sergio Guadarrama, and Trevor Darrell. Caffe: Convolutional Architecture for Fast Feature Embedding. In *ACM Multimedia*, 2014.

[20] Alex Krizhevsky, I. Sutskever, and Geoffrey Hinton. ImageNet classification with deep convolutional neural networks. In *Neural Information Processing Systems*, 2012.

[21] Praveen Kulkarni, Joaquin Zepeda, Frédéric Jurie, Patrick Pérez, and Louis Chevallier. Hybrid multi-layer deep CNN/aggregator feature for image classification. In *International Conference on Acoustics, Speech and Signal Processing*, 2015.

[22] Praveen Kulkarni, Joaquin Zepeda, Frédéric Jurie, Patrick Pérez, and Louis Chevallier. Max-margin, single-layer adaptation of transferred image features. In *BigVision Workshop, Computer Vision and Pattern Recognition*, 2015.

[23] Praveen Kulkarni, Joaquin Zepeda, Frédéric Jurie, Patrick Pérez, and Louis Chevallier. Learning the structure of deep architectures via ℓ_1 penalization. In *British Machine Vision Conference*, 2015.

[24] Svetlana Lazebnik and Cordelia Schmid. Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In *Computer Vision and Pattern Recognition*, 2006.

[25] Julien Mairal, Francis Bach, Andrew Zisserman, and Guillermo Sapiro. Supervised dictionary learning. In *Neural Information Processing Systems*, 2008.

[26] Julien Mairal, Francis Bach, Jean Ponce, and Guillermo Sapiro. Online Dictionary learning for sparse coding. In *International Conference on Machine Learning*, 2009.

[27] Tomasz Malisiewicz, Abhinav Gupta, and Alexei Efros. Ensemble of exemplar-SVMs for object detection and beyond. In *International Conference on Computer Vision*, 2011.

[28] Tomasz Malisiewicz, Abhinav Shrivastava, Abhinav Gupta, and Alexei Efros. Exemplar-SVMs for visual object detection, label transfer and image retrieval. In *International Conference of Machine Learning*, 2012.

[29] Andrew Ng. CS229 Lecture notes. Part V: Support Vector Machines. Technical Report, Stanford University, 2010.

[30] Bruno Olshausen and David Field. Sparse coding with an overcomplete basis set: A strategy employed by V1? *Vision research*, (37)23: 3311–3325, 1997.

[31] Rajat Raina, Alexis Battle, Honglak Lee, Benjamin Packer, and Andrew Ng. Self-taught learning: Transfer learning from unlabeled data. In *International Conference on Machine Learning*, 2007.

[32] Ali Sharif Razavian, Hossein Azizpour, Josephine Sullivan, and Stefan Carlsson. CNN features off-the-shelf: An astounding baseline for recognition. In *Computer Vision and Pattern Recognition Workshops*, 2014.

[33] Shai Shalev-Shwartz and Nathan Srebro. SVM optimization: Inverse dependence on training set size. In *International Conference on Machine Learning*, 2008.

[34] Shai Shalev-Shwartz, Yoram Singer, and Nathan Srebro. Pegasos: Primal estimated sub-gradient solver for SVM. In *International Conference of Machine Learning*, 2007.

[35] P.K. Shivaswamy and Tony Jebara. Relative margin machines. In *Advances in Neural Information Processing Systems*, 2008.

[36] Karl Skretting and Kjersti Engan. Recursive least squares dictionary learning algorithm. *IEEE Transactions on Signal Processing*, 58(4):2121–2130, 2010.

[37] Jianchao Yang, Kai Yu, Yihong Gong, and Thomas Huang. Linear spatial pyramid matching using sparse coding for image classification. In *Computer Vision and Pattern Recognition*, 2009.

[38] Gui-Bo Ye, Yifei Chen, and Xiaohui Xie. Efficient variable selection in support vector machines via the alternating direction method of multipliers. In *International Conference on Artificial Intelligence and Statistics*, 2011.

[39] Joaquin Zepeda and Patrick Pérez. Exemplar SVMs as visual feature encoders. In *Computer Vision and Pattern Recognition*, 2015.

[40] Joaquin Zepeda, Christine Guillemot, and Ewa Kijak. Image compression using sparse representations and the iteration-tuned and aligned dictionary. *IEEE Transactions on Signal Processing*, 5(5):1061–1073, 2011.

[41] Weiyu Zhang, Stella Yu, and Shang-Hua Teng. Power SVM: Generalization with exemplar classification uncertainty. In *Computer Vision and Pattern Recognition*, 2012.

[42] Ji Zhu, Saharon Rosset, Trevor Hastie, and Rob Tibshirani. 1-norm support vector machines. In *Neural Information Processing Systems*, 2003.

[43] Hui Zou and Trevor Hastie. Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 67(2):301–320, 2005.