Learning output kernels with block coordinate descent

Francesco Dinuzzo FDINUZZO@TUEBINGEN.MPG.DE
Max Planck Institute for Intelligent Systems, 72076 Tübingen, Germany

Cheng Soon Ong CHENGSOON.ONG@INF.ETHZ.CH
Department of Computer Science, ETH Zürich, 8092 Zürich, Switzerland

Peter Gehler PGEHLER@MPI-INF.MPG.DE
Max Planck Institute for Informatics, 66123 Saarbrücken, Germany

Gianluigi Pillonetto GIAPI@DEI.UNIPD.IT
Department of Information Engineering, University of Padova, 35131 Padova, Italy

Abstract

We propose a method to learn simultaneously a vector-valued function and a kernel between its components. The obtained kernel can be used both to improve learning performance and to reveal structures in the output space which may be important in their own right. Our method is based on the solution of a suitable regularization problem over a reproducing kernel Hilbert space of vector-valued functions. Although the regularized risk functional is non-convex, we show that it is invex, implying that all local minimizers are global minimizers. We derive a block-wise coordinate descent method that efficiently exploits the structure of the objective functional. Then, we empirically demonstrate that the proposed method can improve classification accuracy. Finally, we provide a visual interpretation of the learned kernel matrix for some well known datasets.

1. Introduction

Classical learning tasks such as binary classification and regression can be solved by synthesizing a real-valued function from the data. More generally, problems such as multiple output regression, multitask learning, multiclass and multilabel classification can be addressed by first learning a vector-valued function, and then applying a suitable transformation to the outputs. In this paper, we introduce a method that simultaneously learns a vector-valued function and the kernel between the components of the output vector. The problem is formulated within the framework of regularization in RKH spaces. We assume that the matrix-valued kernel can be decomposed as the product of a scalar kernel and a positive semidefinite kernel matrix that represents the similarity between the output components, a structure that has been considered in a variety of works, (Evgeniou et al., 2005; Caponnetto et al., 2008; Baldassarre et al., 2010).

In practice, an important issue is the choice of the kernel matrix between the outputs. In some cases, one may be able to fix it by using prior knowledge about the relationship between the components. However, such prior knowledge is not available in the vast majority of the applications, therefore it is important to develop data-driven tools to choose the kernel automatically. The learned kernel matrix can be also used for visualization purposes, revealing interesting structures in the output space. For instance, when applying the method to a multiclass classification problem, the learned kernel matrix can be used to cluster the classes into homogeneous groups. Previous works, such as (Zien & Ong, 2007; Lampert & Blaschko, 2008), have shown that learning convex combinations of kernels on the output space can improve performance.

Our method searches over the whole cone of positive semidefinite matrices. Although the resulting optimization problem is non-convex, we prove that the objective is an invex function (Mishra & Giorgi, 2008), namely it has the property that stationary points are global minimizers. Then, we propose an efficient block-wise coordinate descent algorithm to compute

Appearing in *Proceedings of the 28th International Conference on Machine Learning*, Bellevue, WA, USA, 2011.

an optimal solution. The algorithm alternates between the solution of a so-called discrete time Sylvester equation and a quadratic optimization problem over the cone of positive semidefinite matrices. Exploiting the specific structure of the problem, we propose an efficient solution for both steps.

The four contributions of this paper are as follows. First, we introduce a new problem of learning simultaneously a vector-valued function and the similarity between its components. Second, we show that our non-convex objective is in fact invex, and hence local minimizers are global minimizers. Third, we derive an efficient block-wise coordinate descent algorithm to solve our problem. Finally, we show empirical evidence that our method can improve performances, and also discover meaningful structures in the output space.

2. Spaces of vector-valued functions

We review some facts used to model learning tasks such as multiple output regression, multiclass and multilabel classification. The key idea is to learn functions with values in a vector space.

2.1. Reproducing Kernel Hilbert Spaces

We provide some background here on reproducing kernel Hilbert spaces (RKHSs) of vector-valued functions, i.e. where the outputs are vectors instead of scalars. Let $\mathcal{Y}$ denote an Hilbert space with inner product $\langle \cdot, \cdot \rangle_{\mathcal{Y}}$, and $\mathcal{L}(\mathcal{Y})$ the space of bounded linear operators from $\mathcal{Y}$ into itself. The following definitions introduce positive semidefinite $\mathcal{Y}$-kernel and RKHS of $\mathcal{Y}$-valued functions, that extend the classical concepts of positive semidefinite kernel and RKHS. See Micchelli & Pontil (2005); Carmeli et al. (2006) for further details.

Definition 2.1 (Positive semidefinite $\mathcal{Y}$-kernel)**.** *Let $\mathcal{X}$ denote a non-empty set and $\mathcal{Y}$ an Hilbert space. A symmetric function $H : \mathcal{X} \times \mathcal{X} \to \mathcal{L}(\mathcal{Y})$ is called* positive semidefinite $\mathcal{Y}$-kernel on $\mathcal{X}$ *if, for any finite integer ℓ, the following holds*

$$\sum_{i=1}^{\ell} \sum_{j=1}^{\ell} \langle y_i, H(x_i, x_j) y_j \rangle_{\mathcal{Y}} \geq 0, \qquad \forall (x_i, y_i) \in (\mathcal{X}, \mathcal{Y}) \,.$$

We now introduce Hilbert spaces of vector-valued functions that can be associated to positive semidefinite $\mathcal{Y}$-kernels.

Definition 2.2 (RKHS of $\mathcal{Y}$-valued functions)**.** *A Reproducing Kernel Hilbert Space (RKHS) of $\mathcal{Y}$-valued functions $g : \mathcal{X} \to \mathcal{Y}$ is an Hilbert space $\mathcal{H}$ such that, for all $x \in \mathcal{X}$, there exists $C_x \in \mathbb{R}$ such that*

$$\|g(x)\|_{\mathcal{Y}} \leq C_x \|g\|_{\mathcal{H}}, \qquad \forall g \in \mathcal{H}.$$

It turns out that every RKHS of $\mathcal{Y}$-valued functions $\mathcal{H}$ can be associated with a unique positive semidefinite $\mathcal{Y}$-kernel H, called the *reproducing kernel*. Conversely, given a positive semidefinite $\mathcal{Y}$-kernel H on $\mathcal{X}$, there exists a unique RKHS of $\mathcal{Y}$-valued functions defined over $\mathcal{X}$ whose reproducing kernel is H. The standard definition of positive semidefinite (scalar) kernel and RKHS (of real-valued functions) can be recovered by letting $\mathcal{Y} = \mathbb{R}$.

For the rest of the paper, we assume $\mathcal{Y} = \mathbb{R}^m$. In such a case, $\mathcal{L}(\mathcal{Y})$ is just the space of square matrices of order m. By fixing a basis $\{b_i\}_{i \in \mathcal{T}}$ for the output space, where $\mathcal{T} = \{1, \ldots, m\}$, one can uniquely define an associated (scalar-valued) kernel R over $\mathcal{X} \times \mathcal{T}$ such that

$$\langle b_i, H(x_1, x_2) b_j \rangle_{\mathcal{Y}} = R((x_1, i), (x_2, j)),$$

so that an $\mathcal{Y}$-kernel can be seen as a function that maps two inputs into the space of square matrices of order m. Similarly, by fixing any function $g : \mathcal{X} \to \mathcal{Y}$, one can uniquely define an associated function $h : \mathcal{X} \times \mathcal{T} \to \mathbb{R}$ such that

$$g(x) = \sum_{i \in \mathcal{T}} h(x, i) b_i.$$

Consequently, a space of $\mathcal{Y}$-valued functions over $\mathcal{X}$ is isomorphic to a standard space of scalar-valued functions defined over the input set $\mathcal{X} \times \mathcal{T}$. In conclusion, by fixing a basis for the output space, the problem of learning a vector-valued function can be equivalently regarded as a problem of learning a scalar function defined over an enlarged input set, a modeling approach that is also used in the structured output learning setting (Bakir et al., 2007).

2.2. Learning Tasks

As mentioned in the introduction, many learning tasks can be solved by first learning a function taking values in $\mathbb{R}^m$. We list here brief descriptions of some commonly used models. First of all, in multiple output regression, each component of the vector directly corresponds to an output. For multiclass classification, output data are modeled as binary vectors, with $+1$ in the position corresponding to the class. By interpreting the components of the learned vector-valued function g as "confidence scores" for the different classes, one can consider a classification rule of the form $\widehat{y}(x) = \arg\max_{i \in \mathcal{T}} g_i(x)$. For binary multilabel classification, outputs are vectors with ± 1 at the different components. The final classification rule is given by the component-wise application of a sign function: $\widehat{y}_i(x) = \operatorname{sign}(g_i(x))$.

3. Learning an output kernel

In this section, we introduce and study an optimization problem that can be used to learn simultaneously a vector-valued function and a kernel on the outputs.

3.1. Matrix notation

Let $\mathbb{S}^m_{++} \subseteq \mathbb{S}^m_+ \subseteq \mathbb{M}^m$ denote the open cone of positive definite matrices, the closed cone of positive semidefinite matrices, and the space of square matrices of order m, respectively. For any $\mathbf{A}, \mathbf{B} \in \mathbb{M}^m$, denote the Frobenius inner product $\langle \mathbf{A}, \mathbf{B} \rangle_F := \operatorname{tr}(\mathbf{A}^T\mathbf{B})$, and the induced norm $\|\mathbf{A}\|_F := \sqrt{\langle \mathbf{A}, \mathbf{A} \rangle_F}$. We denote the identity matrix as $\mathbf{I}$ and, for any matrix $\mathbf{A}$, let $\mathbf{A}^T$ denote its transpose, and $\mathbf{A}^\dagger$ its Moore-Penrose pseudoinverse. The Kronecker product is denoted by $\otimes$, and the vectorization operator by $\operatorname{vec}(\cdot)$. In particular, for matrices $\mathbf{A}, \mathbf{B}, \mathbf{C}$ of suitable size, we use the identity $\operatorname{tr}(\mathbf{A}^T\mathbf{B}\mathbf{A}\mathbf{C}) = \operatorname{vec}(\mathbf{A})^T(\mathbf{C}^T \otimes \mathbf{B})\operatorname{vec}(\mathbf{A})$.

3.2. Regularized risk

In the following, we assume $\mathcal{Y} = \mathbb{R}^m$. Let $\mathcal{H}$ denote the RKHS of $\mathcal{Y}$-valued functions $g : \mathcal{X} \to \mathcal{Y}$ associated with the kernel H defined as

$$H = K \cdot \mathbf{L},$$

where K is a positive semidefinite scalar kernel on $\mathcal{X}$ that measures the similarity between inputs, and $\mathbf{L} \in \mathbb{S}^m_+$ is a symmetric positive semidefinite matrix that encodes the relationships between the output's components.

Our method simultaneously learns a function $g \in \mathcal{H}$ and the matrix $\mathbf{L}$ (output kernel) from a set of ℓ input-output data pairs $(x_i, y_i) \in \mathcal{X} \times \mathcal{Y}$. We adopt a regularization approach where a penalty on the function g and on the output kernel matrix $\mathbf{L}$ is used to suitably constrain the solution and make the problem well posed:

$$\min_{\mathbf{L} \in \mathbb{S}^m_+} \left[\min_{g \in \mathcal{H}} \left(\sum_{i=1}^{\ell} \frac{\|g(x_i) - y_i\|_2^2}{2\lambda} + \frac{\|g\|_{\mathcal{H}}^2}{2} + \frac{\|\mathbf{L}\|_F^2}{2} \right) \right]. \tag{1}$$

Note that the objective functional contains only one regularization parameter. A second regularization parameter is not needed as it can be shown to be equivalent to rescaling K. According to the representer theorem for vector-valued functions (Micchelli & Pontil, 2005), the inner minimization problem admits an optimal solution of the form

$$g^*(x) = \sum_{i=1}^{\ell} H(x, x_i) c_i = \mathbf{L} \sum_{i=1}^{\ell} c_i K(x, x_i), \tag{2}$$

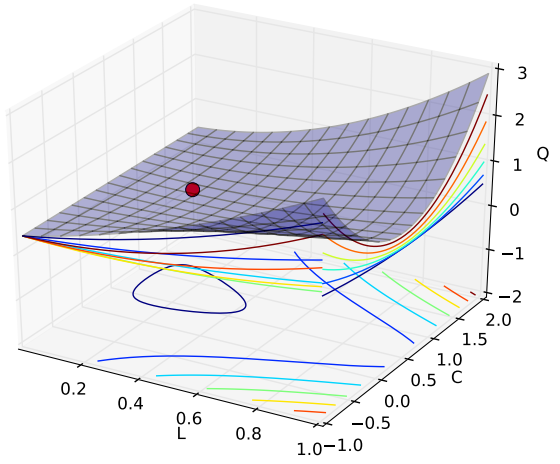


Figure 1. The objective function for scalar $\mathbf{K}$ and $\mathbf{L}$. Although the level sets are non-convex, stationary points are also global minimizers.

where $c_i \in \mathcal{Y}$ $(i = 1, \ldots, \ell)$ are suitable vectors. Let $\mathbf{K} \in \mathbb{S}^\ell_+$ be such that $\mathbf{K}_{ij} = K(x_i, x_j)$, and $\mathbf{Y}, \mathbf{C} \in \mathbb{R}^{\ell \times m}$ such that

$$\mathbf{Y} = (y_1, \ldots, y_\ell)^T, \qquad \mathbf{C} = (c_1, \ldots, c_\ell)^T.$$

By plugging the expression (2) of g into the objective functional of problem (1), we obtain

$$\min_{\mathbf{L} \in \mathbb{S}^m_+} \min_{\mathbf{C} \in \mathbb{R}^{\ell \times m}} Q(\mathbf{L}, \mathbf{C}),$$

where

$$Q(\mathbf{L}, \mathbf{C}) := \frac{\|\mathbf{Y} - \mathbf{K}\mathbf{C}\mathbf{L}\|_F^2}{2\lambda} + \frac{\langle \mathbf{C}^T\mathbf{K}\mathbf{C}, \mathbf{L} \rangle_F}{2} + \frac{\|\mathbf{L}\|_F^2}{2}. \tag{3}$$

Observe that Q is strongly convex with respect to $\mathbf{L}$ and convex with respect to $\mathbf{C}$. However, Q is not (quasi)-convex with respect to the pair $(\mathbf{L}, \mathbf{C})$. This can be seen from Figure 1, by noticing that the level sets of the objective functional in the two-dimensional case $(m = \ell = 1)$ are non-convex. Nevertheless, we will show in Theorem 3.3 that the objective has the property that local minimizers are global minimizers. Furthermore, we derive an efficient optimization method in Section 4.

3.3. Invex functions

Recall that a function f defined over a convex set is called quasiconvex if all its sublevel sets are convex. Although the function Q defined in (3) is not quasiconvex (and hence also not convex), it turns out that it is a so-called invex function, so that stationary points are also global minimizers. We now review some basic

facts about invex functions. For more details, we refer to (Mishra & Giorgi, 2008).

Definition 3.1 (Invex function)**.** *Let $\mathcal{A}$ denote an open set. A differentiable function $f : \mathcal{A} \to \mathbb{R}$ is called invex if there exists a function $\eta : \mathcal{A} \times \mathcal{A} \to \mathbb{R}^n$ such that*

$$f(x_1) - f(x_2) \geq \eta(x_1, x_2)^T \nabla f(x_2), \quad \forall x_1, x_2 \in \mathcal{A}.$$

The definition of an invex function is a generalization of the gradient inequality for differentiable convex functions, which corresponds to $\eta(x_1, x_2) = x_1 - x_2$. Remarkably, invex functions can be characterized as follows (Craven & Glover, 1985; Ben-Israel & Mond, 1986).

Theorem 3.2. *A differentiable function $f : \mathcal{A} \to \mathbb{R}$ is invex if and only if every stationary point is a global minimizer.*

While differentiable convex functions are particular instances of invex functions, the class of invex functions is much larger. The class of quasi-convex functions partially overlaps with the class of invex functions. For instance, the function $f_{QC}(x) = x^3$ is quasi-convex but not invex, while the function $f_I(x_1, x_2) = 1 + x_1^2 - e^{-x_2^2}$ is invex but not quasi-convex. For these and other relationships between invex functions and other classes of functions, see (Mishra & Giorgi, 2008).

3.4. The objective is invex

With the definition of invex functions above, we can now state the following result.

Theorem 3.3. *Functional Q defined as in (3) is an invex function over the open set $\mathbb{S}^m_{++} \times \mathbb{R}^{\ell \times m}$. In particular, local minimizers are also global minimizers.*

Proof of Theorem 3.3. First of all, let

$$F(\mathbf{L}, \mathbf{Z}) = \frac{\|\mathbf{Y} - \mathbf{Z}\|_F^2}{2\lambda} + \frac{\|\mathbf{L}\|_F^2}{2} + h(\mathbf{L}, \mathbf{Z}),$$

where

$$h(\mathbf{L}, \mathbf{Z}) = \begin{cases} \langle \mathbf{Z}^T \mathbf{K}^\dagger \mathbf{Z}, \mathbf{L}^{-1} \rangle_F / 2, & \mathbf{Z} = \mathbf{KCL} \\ +\infty, & \text{otherwise} \end{cases}$$

and observe that, by the properties of the trace and the pseudo-inverse, we have

$$Q(\mathbf{L}, \mathbf{C}) = F(\mathbf{L}, \mathbf{KCL}). \tag{4}$$

Now, we show that F is jointly convex in $\mathbf{L}$ and $\mathbf{Z}$. Since the first two terms are quadratic an thus jointly convex, we only need to prove convexity of $h(\mathbf{L}, \mathbf{Z})$. Let

$$\mathbf{H} = \mathbf{L} \otimes \mathbf{K}, \qquad z = \mathrm{vec}(\mathbf{Z}), \tag{5}$$

and observe that it suffices to prove convexity in the effective domain of h, where

$$\mathbf{H}\mathbf{H}^\dagger z = z.$$

Now, we also have

$$\langle \mathbf{Z}^T \mathbf{K}^\dagger \mathbf{Z}, \mathbf{L}^\dagger \rangle_F = z^T \mathbf{H}^\dagger z,$$

and it turns out that this last function is jointly convex with respect to z and $\mathbf{H}$. Indeed, by the generalized Schur complement lemma (Albert, 1969), we have

$$z^T \mathbf{H}^\dagger z \leq \alpha \quad \Leftrightarrow \quad \begin{pmatrix} \mathbf{H} & z \\ z^T & \alpha \end{pmatrix} \in \mathbb{S}^{m+1}_+,$$

so that the epigraph is convex. Since $(\mathbf{H}, z)$ is a linear function of $(\mathbf{L}, \mathbf{Z})$, the functional is also convex with respect to these variables. In conclusion, h and hence F are jointly convex.

Now, let $(\mathbf{L}_*, \mathbf{C}_*)$ denote a stationary point of Q where $\mathbf{L}$ is positive definite, and set $\mathbf{Z}_* = \mathbf{K}\mathbf{C}_*\mathbf{L}_*$. In view of (4), it follows that

$$\begin{aligned} \mathbf{0} &= \frac{\partial Q}{\partial \mathbf{L}}(\mathbf{L}_*, \mathbf{C}_*) = \frac{\partial F}{\partial \mathbf{L}}(\mathbf{L}_*, \mathbf{Z}_*) + \mathbf{C}_*^T \mathbf{K} \frac{\partial F}{\partial \mathbf{Z}}(\mathbf{L}_*, \mathbf{Z}_*) \\ \mathbf{0} &= \frac{\partial Q}{\partial \mathbf{C}_*}(\mathbf{L}_*, \mathbf{C}_*) = \mathbf{K} \frac{\partial F}{\partial \mathbf{Z}_*}(\mathbf{L}_*, \mathbf{Z}_*) \mathbf{L}_* \end{aligned}$$

In particular, let

$$\mathbf{S} := \frac{\partial F}{\partial \mathbf{Z}}(\mathbf{L}_*, \mathbf{Z}_*) = \mathbf{Y} - \lambda \mathbf{C}_* - \mathbf{Z}_*,$$

and let $\mathbf{U} \in \mathbb{R}^{\ell \times m}$ denote the matrix whose columns are obtained by projecting the columns of $\mathbf{S}$ over the nullspace of $\mathbf{K}$. Then, letting

$$\widetilde{\mathbf{C}} = \mathbf{C}_* - \mathbf{U}/\lambda, \qquad \widetilde{\mathbf{Z}} = \mathbf{K}\widetilde{\mathbf{C}}\mathbf{L}_*,$$

we have that $(\mathbf{L}_*, \widetilde{\mathbf{C}})$ is still stationary for Q, and in addition

$$Q(\mathbf{L}_*, \mathbf{C}_*) = Q(\mathbf{L}_*, \widetilde{\mathbf{C}}) = F(\mathbf{L}_*, \widetilde{\mathbf{Z}}).$$

Furthermore, the columns of $\left(\mathbf{Y} - \lambda\widetilde{\mathbf{C}} - \widetilde{\mathbf{Z}}\right)$ belong to the range of $\mathbf{K}$, therefore we also have

$$\mathbf{0} = \frac{\partial F}{\partial \mathbf{Z}}(\mathbf{L}_*, \widetilde{\mathbf{Z}}) = \mathbf{Y} - \lambda\widetilde{\mathbf{C}} - \widetilde{\mathbf{Z}}.$$

As a consequence, it also follows

$$\mathbf{0} = \frac{\partial F}{\partial \mathbf{L}}(\mathbf{L}_*, \widetilde{\mathbf{Z}}).$$

In conclusion, the pair $(\mathbf{L}_*, \widetilde{\mathbf{Z}})$ is a stationary point for F and, since F is convex, also a global minimizer. Hence, we have

$$Q(\mathbf{L}_*, \mathbf{C}_*) = F(\mathbf{L}_*, \widetilde{\mathbf{Z}}) \leq F(\mathbf{L}, \mathbf{Z}),$$

for any $(\mathbf{L}, \mathbf{Z})$, in particular for $\mathbf{Z} = \mathbf{KCL}$. In view of (4), it follows that $(\mathbf{L}_*, \mathbf{C}_*)$ is a global minimizer of Q. Finally, it follows by Theorem 3.2 that Q is invex. □

4. A block coordinate descent method

In the following, we derive a block-wise coordinate descent technique (Algorithm 1) for minimizing the functional of equation (3). The algorithm alternates between minimization with respect to $\mathbf{L}$ and minimization with respect to $\mathbf{C}$. The key computational steps are in lines 3 and 6, which are derived in Section 4.1 and 4.2 respectively.

4.1. Sub-problem w.r.t. coefficient matrix C

From Equation (3) we see that for any fixed $\mathbf{L}$, $\mathbf{C}$ is optimal if and only if

$$\mathbf{0} = \frac{\partial Q(\mathbf{L}, \mathbf{C})}{\partial \mathbf{C}} = -\frac{\mathbf{K}(\mathbf{Y} - \lambda\mathbf{C} - \mathbf{KCL})\mathbf{L}}{\lambda}.$$

Hence, a sufficient condition for $\mathbf{C}$ to be optimal is

$$\mathbf{Y} - \lambda\mathbf{C} - \mathbf{KCL} = \mathbf{0}, \tag{6}$$

a linear matrix equation called discrete-time Sylvester equation (Sima, 1996). In principle, this can be rewritten and solved as a large linear system of equations of the form

$$(\mathbf{L} \otimes \mathbf{K} + \lambda\mathbf{I})\mathrm{vec}(\mathbf{C}) = \mathrm{vec}(\mathbf{Y}).$$

However, this naive solution is not efficient for large scale problems. Sylvester equations arise in control theory, and there exist many efficient techniques that take advantage of the structure. We use an algorithm implemented in SLICOT [1].

4.2. Sub-problem w.r.t. output kernel L

For any fixed $\mathbf{C}$, the problem with respect to $\mathbf{L}$ is a strongly convex optimization problem in the cone of symmetric positive semidefinite matrices. We derive an update that maintains positive semidefiniteness and can be computed by solving a linear system.

A key observation is the following: if $\mathbf{C}$ has been obtained by solving the Sylvester equation (6), then the positive semidefinite constraint is never active in the sub-problem with respect to $\mathbf{L}$. Namely, we have that

$$\arg\min_{\mathbf{L} \in \mathbb{S}_+^m} Q(\mathbf{L}, \mathbf{C}) = \arg\min_{\mathbf{L} \in \mathbb{M}^m} Q(\mathbf{L}, \mathbf{C}), \tag{7}$$

since the solution of the problem on the right hand side is automatically symmetric and positive semidefinite.

Algorithm 1 Block-wise coordinate descent

1: $\mathbf{L}, \mathbf{C}, \mathbf{E}, \mathbf{Z} \leftarrow \mathbf{0}$
2: **while** $\|\mathbf{Z} + \lambda\mathbf{C} - \mathbf{Y}\|_F \geq \delta$ **do**
3: $\quad \mathbf{C} \leftarrow$ Solution to $\mathbf{KCL} + \lambda\mathbf{C} = \mathbf{Y}$,
4: $\quad \mathbf{E} \leftarrow \mathbf{KC}$
5: $\quad \mathbf{P} \leftarrow \frac{1}{2}\mathbf{E}^T\mathbf{C} - \mathbf{L}$
6: $\quad \mathbf{Q} \leftarrow$ Solution to $\left(\mathbf{E}^T\mathbf{E} + \lambda\mathbf{I}\right)\mathbf{Q} = \mathbf{P}$
7: $\quad \mathbf{L} \leftarrow \mathbf{L} + \lambda\mathbf{Q}$
8: $\quad \mathbf{Z} \leftarrow \mathbf{EL}$
9: **end while**

Lemma 4.1. *Assume that* $\mathbf{C}$ *is a solution of Equation (6) with* $\mathbf{L} = \mathbf{L}_p$ *and let* $\mathbf{E} := \mathbf{KC}$. *If* $\mathbf{L}$ *solves the subproblem at the left hand side of (7), then we have*

$$\mathbf{L} = \mathbf{L}_p + \lambda\mathbf{Q}$$

where $\mathbf{Q}$ *solves the linear system*

$$\left(\mathbf{E}^T\mathbf{E} + \lambda\mathbf{I}\right)\mathbf{Q} = \mathbf{P}, \quad \mathbf{P} := \frac{1}{2}\mathbf{E}^T\mathbf{C} - \mathbf{L}_p. \tag{8}$$

Proof. Observe that matrix $\mathbf{L}$ is optimal for the problem on the right hand side of (7) if and only if

$$\mathbf{0} = \frac{\partial Q(\mathbf{L}, \mathbf{C})}{\partial \mathbf{L}} = -\frac{\mathbf{C}^T\mathbf{K}(\mathbf{Y} - \lambda\mathbf{C}/2 - \mathbf{KCL})}{\lambda} + \mathbf{L},$$

that is

$$\mathbf{L} = \left(\mathbf{C}^T\mathbf{K}^2\mathbf{C} + \lambda\mathbf{I}\right)^{-1}\mathbf{C}^T\mathbf{K}\left(\mathbf{Y} - \frac{\lambda}{2}\mathbf{C}\right). \tag{9}$$

Now, recall that $\mathbf{C}$ satisfies

$$\mathbf{Y} - \frac{\lambda}{2}\mathbf{C} = \frac{\lambda}{2}\mathbf{C} + \mathbf{KCL}_p,$$

where $\mathbf{L}_p$ denote the previous $\mathbf{L}$, which is positive semidefinite. Hence, equation (9) reads

$$\mathbf{L} = \left(\mathbf{C}^T\mathbf{K}^2\mathbf{C} + \lambda\mathbf{I}\right)^{-1}\left(\mathbf{C}^T\mathbf{K}^2\mathbf{CL}_p + \frac{\lambda}{2}\mathbf{C}^T\mathbf{KC}\right), \tag{10}$$

showing that $\mathbf{L}$ is symmetric and positive semidefinite. By letting $\mathbf{E} := \mathbf{KC}$, the update (10) can be rewritten in a more compact form:

$$\begin{aligned}\mathbf{L} &= \left(\mathbf{E}^T\mathbf{E} + \lambda\mathbf{I}\right)^{-1}\left(\mathbf{E}^T\mathbf{EL}_p + \tfrac{\lambda}{2}\mathbf{E}^T\mathbf{C}\right)\\ &= \mathbf{L}_p + \lambda\left(\mathbf{E}^T\mathbf{E} + \lambda\mathbf{I}\right)^{-1}\left(\tfrac{1}{2}\mathbf{E}^T\mathbf{C} - \mathbf{L}_p\right)\\ &= \mathbf{L}_p + \lambda\mathbf{Q}\end{aligned}$$

where (8) holds. □

[1] www.slicot.org

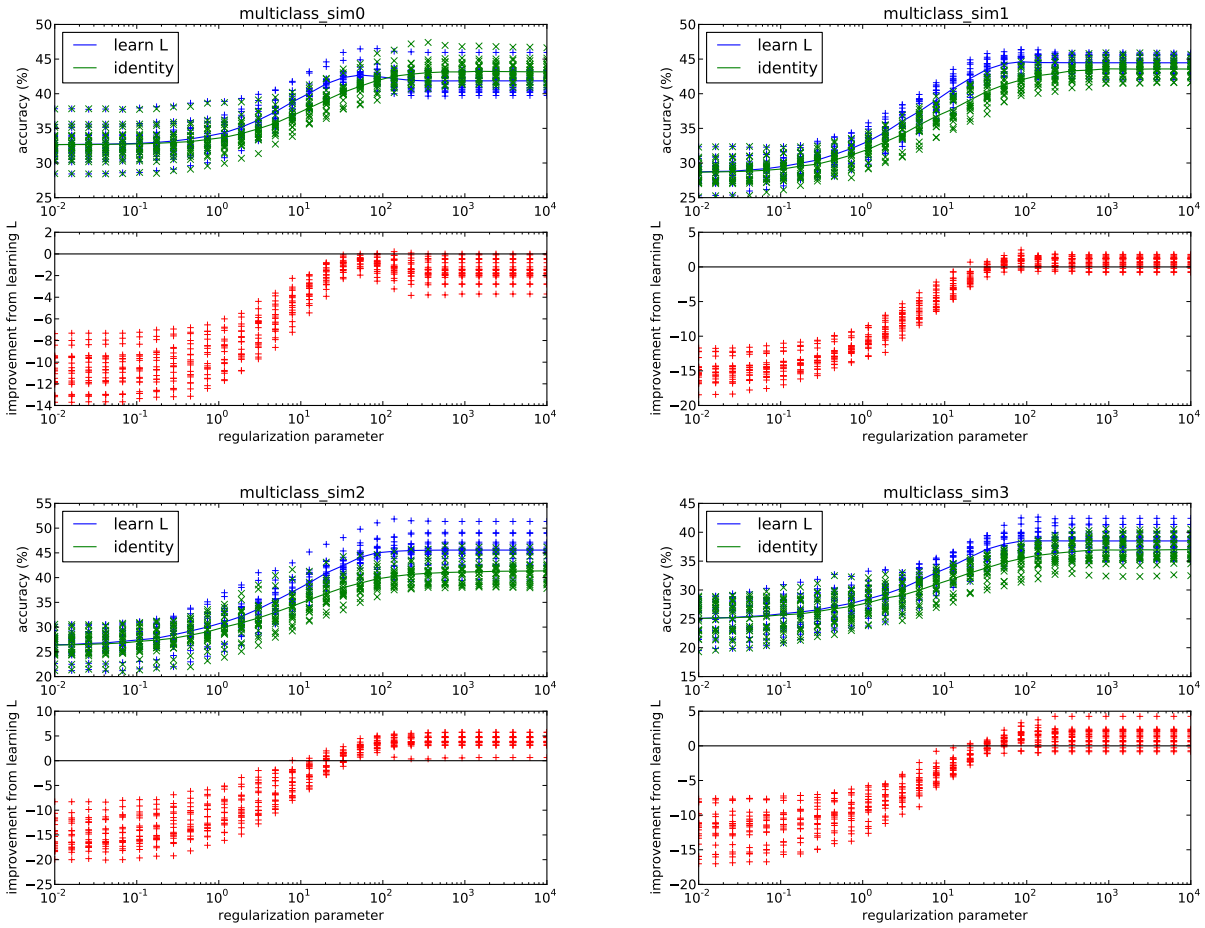


Figure 2. Improvement in classification accuracy from learning the output kernel. The four sub-figures correspond to the four different datasets as described in the text. The top plot in each sub-figure shows the accuracy of the classifier when using the identity kernel (green x) and when using the learned output kernel (blue +). The bottom plots in each subfigure shows the difference between the accuracy of the learned output kernel classifier and the *best standard classifier*. That is, for each split, we find the peak accuracy of the classifier with identity output kernel, and use this as a baseline. The results demonstrate that when there is structure in the labels, our method can utilize this to improve accuracy.

5. Experiments

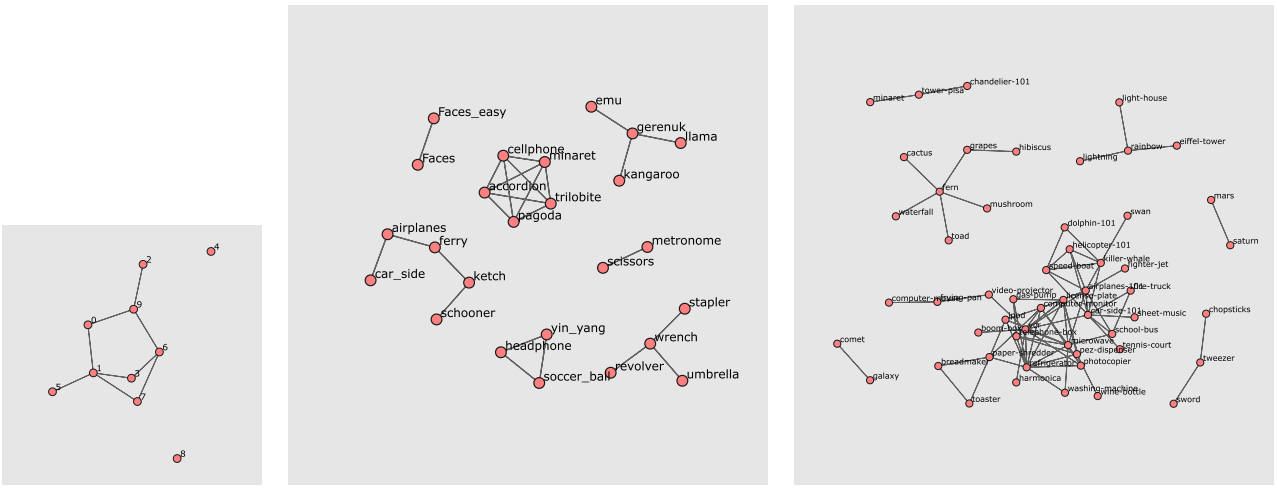
We present two types of empirical evidence showing the usefulness of our method in the multiclass setting[2]. First, we show that learning the class similarities can result in improved accuracies (Section 5.1). Second, we visualize the learned structure between classes for some larger datasets (Section 5.2).

5.1. Improvement in classification accuracy

We simulate an easy structure discovery problem, namely when classes are mistakenly split. This verifies the intuition that, by grouping classes that are really the same, we increase the effective number of examples for each class and hence improve performance. In the experiments, we efficiently compute the solution for several values of the regularization parameter λ by using a warm-start procedure.

We generated data from a 100-dimensional mixture of Gaussians with unit variance. The means of the Gaussians are placed on the simplex centered around the origin, with edges of length $\sqrt{2}$. All datasets have 5 labels, (denoted 1,2,3,4,5) but had varying number of classes. Up to 5 classes were considered (denoted as a,b,c,d,e), with each class corresponding to a Gaussian. The first dataset **sim0** consists of 5 independent classes, and the other 3 datasets were generated with

[2] `people.tuebingen.mpg.de/fdinuzzo/okl.html`



(a) USPS digits (b) Caltech 101 (c) Caltech 256

Figure 3. Interpreting the output kernel, by visualizing edges associated with off diagonal entries of $\mathbf{L}$ with largest values.

the following label structure:

$$\begin{aligned}
\textbf{sim1} \quad & \underbrace{\{0,1\}}_{a}, \underbrace{\{2\}}_{b}, \underbrace{\{3\}}_{c}, \underbrace{\{4\}}_{d} \\
\textbf{sim2} \quad & \underbrace{\{0\}}_{a}, \underbrace{\{1\}}_{b}, \underbrace{\{2,3,4\}}_{c} \\
\textbf{sim3} \quad & \underbrace{\{0\}}_{a}, \underbrace{\{1,2\}}_{b}, \underbrace{\{3,4\}}_{c}
\end{aligned}$$

Each class contains 500 examples, which are then uniformly allocated labels at random when split. For example, **sim1** is a mixture of 4 Gaussians, the first of which contains a 50/50 mix of labels 0 and 1. Note that in this case, even a perfect classifier would only obtain 87.5% accuracy since on a quarter of the examples, it would have 50% accuracy. In 20 different random splits, 5% of the examples were used for training with the linear kernel, and the rest for testing. The small training sets increase the difficulty of the problem, to explore the case when there is barely enough data to train the classifier.

From the results shown in Figure 2, we observe that in all cases with split classes, learning the kernel between labels significantly improve the performance. Since we have to estimate more parameters in our framework, we unsurprisingly perform worse in **sim0**, when the labels are truly independent.

5.2. Interpreting the learned similarity

The output kernel $\mathbf{L}$ provides the full information of distances between classes. We present a visualization of $\mathbf{L}$ by showing the graph whose edges are associated with the largest entries. The visualization was obtained by setting a threshold on the entries of $\mathbf{L}$ to contain only the largest inter class similarities, and than placing the resulting graph in a visually appealing way with Nodebox[3]. In the following experiments, the learned matrices $\mathbf{L}$ are almost diagonal. Performances are comparable with state of the art methods. The experiments on the largest datasets took roughly a day to complete on a standard desktop. The limiting factor was the solution of the Sylvester equation.

5.2.1. USPS digits

We use the standard training/test split of USPS digits, and used a Gaussian kernel of width 40 on the raw pixel representation. The accuracy achieved (96.5%) is comparable to published results. A visualization of the matrix $\mathbf{L}$ containing the top 10 similarities is shown in Figure 3(a). Note that, since we use the Gaussian kernel on the pixels, the local information in the image is lost. It follows that the learned the class similarities are "blind" to shape information. Classes that have correlated "digit" regions are deemed similar.

5.2.2. Caltech image classification

We used the Caltech image classification data (Fei-Fei et al., 2004; Griffin et al., 2007), with 30 examples per class. The similarity between images was computed using average of the kernels used in (Gehler & Nowozin, 2009). This included the PHOG shape descriptor, the SIFT appearance descriptor, region covariance, local binary patterns and local Gabor functions. The results on Caltech 101 and Caltech 256 are

[3] http://nodebox.net/code/index.php/Graph

shown in figure 3(b) and 3(c). In the figures, singleton classes are not shown to avoid clutter. The achieved accuracies (75.3% and 43.9% respectively) are comparable to published results.

In Figure 3(b), we can observe 7 clusters of objects. Some of them are not surprising, such as the one containing Faces and Faces_easy, or the "animal" cluster containing emu, gerenuk, llama and kangaroo. The latter is likely to be due to very similar backgrounds. An interesting cluster, which is fully connected, contains geometric objects with regular substructures, such as cellphone, minaret, accordion, pagoda, and trilobite. From Figure 3(c), we observe several meaningful clusters, such as planets (mars and saturn), space objects (comet and galaxy), sticklike objects (chopsticks, tweezer, sword), and objects of nature (cactus, fern, waterfall, toad, mushroom, grapes, hibiscus). We can conclude that the obtained visualizations are intuitively plausible. In addition, the learned similarity between classes depends both on the similarity between the inputs and the class labels.

6. Conclusions

In the paper, we proposed a method to learn simultaneously a vector-valued function and a kernel between the output's components. The learned kernel encodes structural relationship between the outputs and can be used for visualization purposes.

Our method is formulated as a regularization problem within the framework of RKHS of vector-valued functions. Although the objective functional is nonconvex, we prove that all the stationary points are global minimizers, a property called invexity. By exploiting the specific structure of the problem, we derive an efficient block-wise coordinate descent algorithm that alternates between the solution of a discrete time Sylvester equation and a closed-form update for the output kernel matrix. The method is applied to learning label similarity in a multiclass setting. We can obtain an improvement in classification performance, and visualize the learned structure in the output space.

References

Albert, A. Conditions for positive and nonnegative definiteness in terms of pseudoinverses. *SIAM Journal on Applied Mathematics*, 17(2):434–440, 1969.

Bakir, G., Hofmann, T., Schölkopf, B., Smola, A., Taskar, B., and Vishwanathan, S. V. N. *Predicting Structured Data*. The MIT Press, 2007.

Baldassarre, L., Rosasco, L., Barla, A., and Verri, A. Vector field learning via spectral filtering. In *Machine Learning and Knowledge Discovery in Databases*, volume 6321, pp. 56–71. Springer, 2010.

Ben-Israel, A. and Mond, B. What is invexity? *J. Australian Math. Soc. Ser. B*, 28:1–9, 1986.

Caponnetto, A., Micchelli, C. A., Pontil, M., and Ying, Y. Universal multi-task kernels. *Journal of Machine Learning Research*, 9:1615–1646, 2008.

Carmeli, C., Vito, E. De, and Toigo, A. Vector valued reproducing kernel Hilbert spaces of integrable functions and Mercer theorem. *Analysis and Applications*, 4:377–408, 2006.

Craven, B.D. and Glover, B.M. Invex functions and duality. *Journal of Australian Mathematical Society*, 24:120, 1985.

Evgeniou, T., Micchelli, C. A., and Pontil, M. Learning multiple tasks with kernel methods. *Journal of Machine Learning Research*, 6:615–637, 2005.

Fei-Fei, L., Fergus, R., and Perona, P. Learning generative visual models from few training examples: An incremental bayesian approach tested on 101 object categories. In *CVPR*, pp. 178, 2004.

Gehler, P. and Nowozin, S. On feature combination for multiclass object classification. In *ICCV*, pp. 221–228, 2009.

Griffin, G., Holub, A., and Perona, P. Caltech-256 object category dataset. Technical Report 7694, Caltech, 2007.

Lampert, C.H. and Blaschko, M.B. A multiple kernel learning approach to joint multi-class object detection. In *DAGM*, 2008.

Micchelli, C. A. and Pontil, M. On learning vector-valued functions. *Neural Computation*, 17:177–204, 2005.

Mishra, S. K. and Giorgi, G. *Invexity and optimization*. Nonconvex Optimization and Its Applications. Springer, Dordrecht, 2008.

Sima, V. *Algorithms for Linear-quadratic Optimization*. Marcel Dekker, New York, 1996.

Zien, A. and Ong, C.S. Multiclass multiple kernel learning. In *ICML*, pp. 1191–1198, 2007.