OPTIMIZING KERNEL RIDGE REGRESSION FOR REMOTE SENSING PROBLEMS

Gonzalo Mateo-García, Valero Laparra, Luis Gómez-Chova

Image Processing Laboratory (IPL), University of Valencia, Spain

ABSTRACT
Kernel methods have been very successful in remote sensing problems because of their ability to deal with high dimensional non-linear data. However, they are computationally expensive to train when a large amount of samples are used. In this context, while the amount of available remote sensing data has constantly increased, the size of training sets in kernel methods is usually restricted to few thousand samples. In this work, we modified the kernel ridge regression (KRR) training procedure to deal with large scale datasets. In addition, the basis functions in the reproducing kernel Hilbert space are defined as parameters to be also optimized during the training process. This extends the number of free parameters from two (in the standard KRR with an RBF kernel) to more than fifty thousand in our experiments. The effectiveness of the proposal is illustrated in the problem of surface temperature estimation from MetOp-IASI hyperspectral infrared sounding data. The data set used contains more than one million samples, but the proposed method could potentially be trained with much more data.

Index Terms— kernel ridge regression, KRR, kernel methods, IASI, temperature retrieval, SGD

1. INTRODUCTION

Kernel methods have been very successful in remote sensing problems [1,2], mostly because its ability to deal with high dimensional and highly non-linear problems. For example, they have shown excellent performances on atmospheric parameters retrieval from the 8461 hyperspectral channels of the infrared atmospheric sounding interferometer (IASI) on-board MetOp satellite [3,4]. However, kernel methods are extremely computationally demanding when training on big datasets and therefore models are trained with a limited number of samples and also fitting a limited number of free parameters. In this sense, deep learning models have overshadowed kernel methods in the last years since they can exploit big datasets and are also very flexible due to their great amount of parameters. In this paper, we propose an extension to kernel methods to solve these drawbacks.

Advances in kernel methods for regression include several similar approaches such as support vector regression, kernel ridge regression (KRR), Kriging, Relevance Vector Machines, or Gaussian processes (GPs). All these methods appeared as nonlinear extensions of simple linear regression. Among them, probably the KRR formulation is the simplest one which only substitutes the Gram matrix using the so called kernel function and enforces to minimize the squared error. Hyper-parameters are then fitted using a cross validation (CV) strategy. On the other hand, GPs are formulated using a robust probabilistic framework defining a probabilistic model for each of the involved factors, which allows to train and fit the model parameters using a maximum likelihood (ML) approach.

ML is optimal when the model is well-specified, however, when the probability function is misspecified, CV performs better than ML [6,7]. Although some proposals have modified the GPs framework to optimize the prediction error [8], they still have to make assumptions about the underlying probabilistic model. In this sense, KRR is less restrictive than GPs. However standard CV training strategies are more expensive since usually involve a grid search and, as a consequence, only models with few parameters can be used. Here we propose a solution to take advantage of the simple formulation of KRR while training with large amounts of data, which allows training more complicated models (i.e. with more parameters).

In general, advances in kernel methods for regression such as KRR or GPs have been focused on designing the kernel functions but less effort has been made on selecting properly the training samples that serve as support. While in the case of GPs the family of sparse GPs [9,10] was proposed to alleviate this problem, there is not satisfactory alternative for the KRR framework.

Next section describes the methodology where we first give an introduction on how to use the kernel function to define a regression problem and then we present our proposal. Afterward, an experiment using IASI data is presented to predict surface temperature with different methods.

2. METHODOLOGY

Kernel ridge regression (KRR) formulation is based on extending simple linear regression by substituting the covariance with a kernel function in the dual problem [11]. This allows defining different nonlinear models by just changing the kernel function. An important advantage of KRR is its training simplicity since it only involves finding the parameters that minimize the mean square error. Usually this mini-
mization is performed using a cross validation scheme with a grid search over the parameters space. Although the solution using grid search is robust to local minima, it is extremely expensive and restricts the use of a large number of parameters. A common choice is the RBF kernel and a Tikhonov regularization, which involves fitting just two parameters by CV: the kernel width and the regularization constant, respectively.

In the following subsections, KRR is formally introduced, simple solutions to cope with big data in the context of KRR are presented, and finally our proposed solution is introduced.

2.1. Kernel function

Given a kernel function that measures similarities between points in our input space, \( k(x, x') : \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}^+ \), this kernel function defines a reproducing kernel Hilbert space (rkHs) \([12]\) that we will call \( \mathcal{F} \) by means of applying the Moore-Aronszajn theorem \([13]\). \( \mathcal{F} \) is a space formed by functions of the form \( f(x) = \sum_{i=1}^{\infty} \alpha_i k(x_i, x) \), where \( \sum_{i=1}^{\infty} \alpha_i^2 k(x_i, x_i) < \infty \).

The scalar product between functions in \( \mathcal{F} \) is computed as

\[
\langle f, g \rangle_{\mathcal{F}} = \left( \sum_{i=1}^{\infty} \alpha_i k(x_i, \cdot), \sum_{j=1}^{\infty} \beta_j k(y_j, \cdot) \right)_{\mathcal{F}} = \sum_{i=1}^{\infty} \sum_{j=1}^{\infty} \alpha_i \beta_j k(x_i, y_j). \tag{1}
\]

If \( k \) is a proper kernel the space \( \mathcal{F} \) is formed by all continuous functions \( f : \mathcal{X} \rightarrow \mathbb{R} \).

2.2. Kernel Ridge Regression (KRR)

KRR uses the kernel similarity function to solve a regression problem by defining a least squares cost function with regularization. Given a sample dataset \( \mathcal{D} = \{ x_i, y_i \}_{i=1}^{N} \) describing a regression problem \( (x_i \in \mathcal{X} \rightarrow y_i \in \mathbb{R}) \), KRR aims to solve the problem:

\[
\alpha^* = \arg \min \sum_{i=1}^{N} (f(x_i) - y_i)^2 + \lambda \|f\|_{\mathcal{F}}^2. \tag{2}
\]

The Representer Theorem \([14]\) states that the prediction function, \( f^* \), that minimizes eq. (2) can be represented as:

\[
f^*(x) = \sum_{i=1}^{N} \alpha_i k(x_i, x) = k(x, X) \alpha_{\alpha^*}, \tag{3}
\]

where \( \alpha \) is a vector of weights that have to be optimized. The objective function eq. (2) can be written in matrix form for this \( f^* \) using the definition of similarity in eq. (1):

\[
J(f^*) = \| K_{ff} \alpha - y \|^2 + \lambda \alpha^T K_{ff} \alpha,
\]

where \( K_{ff} \) is a matrix containing the results of the kernel function for the different elements in \( X \). By doing this, we transformed the minimization in the space of functions of the rkHs defined by \( k \) into a minimization problem (linear ridge regression actually) in \( \alpha \in \mathbb{R}^N \). The \( \alpha^* \) that minimizes \( J(f^*) \) is:

\[
\alpha^* = (K_{ff} + \lambda I)^{-1} y. \tag{4}
\]

2.3. Subset of data

KRR has been extremely useful to deal with highly dimensional data since the solution is independent from the input dimension. However, it requires the inversion of a matrix that grows with the number of training samples, eq. (4). In practice, when the amount of samples is large, a subset \( \mathcal{D}_u = \{ x_i, y_i \}_{i=1}^{M} \) of the whole data set, \( \mathcal{D} \), is used. The solution using only \( \mathcal{D}_u \) is then:

\[
f^*_u(x) = \sum_{i=1}^{M} \alpha_i k(x_i, x) = k(x, X_u) \alpha_u. \tag{5}
\]

This solution is called subset of data (SoD) in the GPs framework, \([9]\). By choosing \( M << N \) the computational load is significantly reduced:

\[
\alpha_u^* = (K_{uu} + \lambda I)^{-1} y \tag{6}
\]

2.4. Optimizing the basis in KRR

The SoD method (sec. 2.3) proposes a very simple solution to reduce the computational load of the KRR by reducing the amount of data used in the training procedure. However, note that this subset does not contain the information from all available samples. This is a problem when \( M \) is much smaller than \( N \), since \( X_u \) and \( \alpha^*_u \) are not going to be rich enough to represent all the information in \( \mathcal{D} \). There exist some proposals in the literature to exploit some of the information available in all data to obtain the weights \( \alpha_u \) and the basis \( X_u \). On the one hand, in order to optimize the weights \( \alpha_u \) using all data, perhaps the most relevant is the Nyström solution \([15]\), and its counterpart in the GPs framework is the subset of regressors (SoR) \([9]\). On the other hand, the optimization of the basis \( X_u \), in the KRR framework, is only approached from the discrete optimization point of view; that is: finding \( X_u \subset X \) using some heuristic criteria \([16][17]\). Here we present a formulation in the KRR framework, which optimizes both the weights \( \alpha_u \) and basis \( X_u \). It consists in relaxing the criteria \( X_u \subset X \), which allows learning from the whole data set while maintaining tractability.

2.4.1. Problem formulation

We will define \( f^* \) depending on a set of \( M \) points \( X_u = \{ x_{ui} \}_{i=1}^{M} \), however, they are not restricted to \( X_u \subset X \). Now,
\( X_u \) are free variables in the input space:

\[
f^*(x) = \sum_{i=1}^{M} \alpha_{ui} k(x_{ui}, x) = \frac{1}{M} \sum_{i=1}^{M} \alpha_{ui},
\]

and the risk \( J \) is:

\[
J(\alpha_u, X_u, \theta_u) = \|K_{fu}(\theta_u) - y\|^2 + \lambda \alpha_u^\top K_{uu} \alpha_u
\]

where \( \theta_u \) are the kernel parameters.

Our proposal consists in minimizing this risk jointly in all parameters: \( \alpha_u, X_u \) and \( \theta_u \) using gradient based optimization methods. In particular, we consider iterative algorithms where we start with an initial solution \( \alpha_u(0), X_u(0) \) and \( \theta_u(0) \) that is improved at each iteration using gradient descent like algorithms.

Among gradient-based optimization algorithms the stochastic gradient descent (SGD) family has become extremely popular in machine learning applications. This is mainly because computing the exact gradient of a risk function like (7) is expensive, whereas computing an stochastic approximation of the gradient is much cheaper. To see this, consider that the risk function in equation (7) is a sum over all training pairs in \( D \); therefore, the computational cost of computing the exact gradient depends on the size of \( D \): \( N \). On the other hand, an stochastic gradient uses only an small random sample of \( D \) to compute the gradient at each step. This drops the dependence on \( N \) of the computational complexity of the gradient which makes the algorithm appropriate for big data scenarios. In this way, we propose to use SGD algorithms to optimize the risk function (7). In addition, for the initialization of the parameters, we propose to use the SoD solution of section 2.3.

By doing this the algorithm starts with a reasonable solution that will be iteratively improved using all the available data. In all experiments parameters are fitted using stochastic gradient descent with mini batches of forty thousand samples. We use the ADAM momentum method [18].

2.4.2. Parameters gradient

In order to implement the proposed solution we need the expression of the gradients of the risk function (7). In the case of the basis \( X_u \) and the parameters of the kernel \( \theta_u \), these gradients highly depend on the selected kernel function. We will restrict ourselves to the RBF kernel, which is the one used in the experiments.

**Basis \( X_u \):** Derivatives of the kernel with respect to the basis \( X_u \) are

\[
\nabla_{X_u} k(x, X_u) = -\frac{1}{\sigma^2} (x^j - x^j_u) k(x, X_u),
\]

where the superindex makes reference to the dimension. Derivatives of the risk (7) w.r.t. the basis follows using the chain rule.

**Weights \( \alpha_u \):** Derivatives of the risk \( J \) with respect to the weights \( \alpha_u \) are

\[
\nabla_{\alpha_u} J = K_{fu}(\theta_u) \top (K_{fu}(\theta_u) \alpha_u - y)
\]

**Kernel parameters \( \theta_u \):** Again we first need the derivative of the kernel function with respect to the parameters \( \nabla_{\theta_u} k(x, X_u) \). Which is for the case of the RBF kernel

\[
\nabla_{\theta_u} k(x, X_u) = \frac{\|x^j - x^j_u\|^2}{2\sigma^2} k(x, X_u)
\]

Derivative of the risk \( J \) (7) w.r.t. the kernel parameters follows again using the chain rule.

3. EXPERIMENTAL RESULTS

The infrared atmospheric sounding interferometer (IASI) instrument on-board the MetOp polar orbiting satellite series provides measurements with high spectral resolution, which its primary purpose is to improve weather predictions and climate models. IASI covers the spectral range between 645 and 2760 cm\(^{-1}\) yielding 8461 spectral channels. The instrument provides a global Earth coverage every 12 hours, representing 14 orbits in a sun-synchronous mid-morning orbit. Our target variable is the surface temperature which is obtained from the ERA Interim model from the European Center for Medium-Range Weather Forecast. In the experiment, we used twelve orbits: six for training and six for testing the model. Each orbit consists of around hundred thousand hyperpixels.

We first apply Principal Components Analysis to reduce the spectral dimensionality to fifty dimensions. We then compare three different methods: linear regression (LR), KRR trained using an standard grid-search cross-validation scheme, and KRR using the proposed method. For both KRR methods we used five hundred basis. In the training step we used 50.000 samples for the classical KRR and 600.000 for the proposed approach and a linear regression.

Figure 1 shows the results for the six test orbits. For the three methods the errors are distributed in clusters. The LR clusters are more scattered, while the KRR are more localized. The proposed approach provides more homogeneous retrievals with more localized errors and large regions with smaller errors. These results are consistent with the global RMS errors: 6.22 K for LR, 5.67 K for KRR, and 4.95 K for our proposal. Note that improvement between KRR and LR is roughly the same as the one between our proposal and KRR.

4. CONCLUSIONS

We proposed a modification of the KRR method that allows training the model using a large amount of samples. The approach is based on the parameters optimization using gradient descent framework instead of the classical grid search. This allows to optimize much more complex models with a huge amount of parameters. In particular, we proposed to optimize
the basis functions employed in the kernel instead of using a subset as it is usually done.

The method can be easily trained with stochastic gradient descent, which allows to potentially use millions of samples. Results show that the performance is clearly increased with respect to the classical KRR. Further work will be focused on performing experiments in other databases as well as an exhaustive comparison with large scale methods like sparse GPs, Nyström based approaches, and random basis methods.

5. REFERENCES


