Transform Learning Based Function Approximation for Regression and Forecasting

Kriti Kumar[1], Angshul Majumdar[2], M Girish Chandra[1], and A Anil Kumar[1]

[1] TCS Research and Innovation, Bangalore, India
{kriti.kumar, m.gchandra, achannaanil.kumar}@tcs.com
[2] IIIT Delhi, New Delhi, India
angshul@iiitd.ac.in

Abstract. Regression and forecasting can be viewed as learning the functions with the appropriate input and output variables from the data. To capture the complex relationship among the variables, different techniques like, kernelized dictionary learning are being explored in the existing literature. In this paper, the transform learning based function approximation is presented which has computational and performance advantages over dictionary based techniques. Apart from providing the formulation and derivation of the necessary update steps, the performance results obtained with both synthetic and real data are presented in the paper. The initial results obtained with both the basic and kernelized versions demonstrate the usefulness of the proposed technique for regression and forecasting tasks.

Keywords: Transform learning · Dictionary learning · Kernel methods · Regression.

1 Introduction

We are living in the era of massive data generation from different entities around us; for instance, the huge amount of data being emanated from the Internet of Things (IoT). The major applications of which lie in the consumer, commercial, industrial and infrastructure sectors. Major challenges faced by any IoT application is to clean, process and make requisite inferences from the vast amount of data acquired by the sensors. Classification and regression are two important facets of inference. Both of these can be viewed as function approximation problem with appropriate input and output variables. For instance, in classification, the input variable is a data vector and the output corresponds to the class label. In regression, the output variable is continuous in nature and the function captures the relationship of the output with the input variables. With the abundance of data at our disposal, the trend is to derive the said function approximation from the data (data-driven).

It is well known that regression plays an important role in a lot of applications like, finding causal relationship between variables in biological systems, weather

data analysis, market research studies, customer survey results, fine-tuning manufacturing and delivery processes etc. Regression models are functions depicting the regression which can range from simple to complex models. This function approximation interpretation also enables the time series forecasting to be viewed as a regression problem. In all these applications, getting an accurate model which can represent the data well is of prime importance. For this purpose, various signal processing based data representation models using dictionaries and transforms have been explored in literature [1], [2], [3], [4], [5], [6], [7], [8].

Dictionaries learnt from the data are actively researched in the last decade in the signal processing community for both synthesis and analysis problems [9], [10], [11], [12]. The synthesis dictionary based models especially those involving sparsity are more popular having many successful applications like, denoising, inpainting, image super resolution, compressed sensing (MRI, CT), classification, etc. The basic formulation of synthesis dictionary learning is given as [1]:

$$\boldsymbol{X} = \boldsymbol{D}\boldsymbol{Z} \tag{1}$$

where, $\boldsymbol{X} \in \mathbb{R}^{L \times N}$ is the data that is represented by the learnt dictionary $\boldsymbol{D} \in \mathbb{R}^{L \times K}$ containing K atoms as its columns and the learnt coefficients $\boldsymbol{Z} \in \mathbb{R}^{K \times N}$.

Recently, the traditional and the kernelized dictionary versions [13] have also been used for learning data representations to facilitate regression problems [14], [15]. There are two different ways in which they are utilized for machine learning tasks: (i) as a two stage approach where, the dictionary coefficients are learnt in the first stage and then fed as features for machine learning based classifiers/regressors in the second stage [16]; (ii) as a single stage approach where, the features and classification/regression weights are learnt together in a joint optimization framework [14], [15], [17]. The single stage approach has a better performance compared to two stage approach since, the output label/variable is utilized effectively while learning the dictionary and the associated coefficients. Although being successful, the sparse coding solved repeatedly for dictionary learning is NP-hard and the approximate synthesis sparse coding algorithms can be computationally expensive. Moreover, the dictionary learning problem is highly non-convex, and there is a high chance of algorithms getting stuck in bad local minima. To address these problems, transform learning based techniques have gained more importance.

Transform learning is an analysis approach, where the data is analyzed by learning a transform to produce the associated coefficients. Unlike dictionary which is an inverse learning problem, transform is a forward learning problem. In signal processing literature, it is well known that transform learning has an advantage over dictionaries in terms of application scenarios, accuracy and complexity [5], [6], [7], [18]. Especially in the image domain, transform learning methods produces state-of-the-art results [6], [19], [20].

Motivated by the low complexity and performance advantages of transform learning, in this paper, an attempt has been made to formulate the regression problem using transform learning framework. A joint optimization is carried out

to learn the transform, associated coefficients and regression weights together. Formulations for both, the basic (Transform Learning for Regression (TLR)) and the kernelized versions (Kernel Transform Learning for Regression (KTLR)) of the transform are proposed here. To the best of our knowledge, these techniques have not been explored so far in the literature. Viewing regression as function approximation problem, the scope of the proposed techniques have been extended to the important problem of time-series forecasting. Extensive study on different datasets are presented along with comparisons with both (basic and kernel) dictionary versions (Dictionary Learning for Regression (DLR), Kernel Dictionary Learning for Regression (KDLR), Kernel Regression (KR) and Linear Regression (LR). The results demonstrate the potential of the proposed techniques for regression.

Towards providing the necessary details of the proposed techniques, the rest of the paper is organized as follows. Section 2 presents in brief, the formulation of basic and kernel transform learning techniques. This is followed by Section 3 which gives the details of the proposed methodology for regression using both basic and kernel transform learning methods. Section 4 provides the experimental results obtained with synthetic and real data along with the performance comparisons with other methods. Finally, Section 5 concludes the work.

2 Introductory aspects related to Transform Learning

This section presents a brief background on both the basic and kernel versions of transform learning to set the stage for the proposed formulation for regression.

2.1 Basic Transform Learning

Transform learning is an analysis approach for data representation whose basic formulation is given as [7]:

$$\boldsymbol{TX} = \boldsymbol{Z} \tag{2}$$

where, $\boldsymbol{X} \in \mathbb{R}^{L\times N}$ is the data matrix, $\boldsymbol{T} \in \mathbb{R}^{K\times L}$ is the transform and $\boldsymbol{Z} \in \mathbb{R}^{K\times N}$ are the coefficients.

Ravishankar and Bresler [5], formulated the transform learning problem as:

$$\min_{\boldsymbol{T},\boldsymbol{Z}} \|\boldsymbol{TX} - \boldsymbol{Z}\|_F^2 + \lambda(\|\boldsymbol{T}\|_F^2 - \log\det \boldsymbol{T}) + \mu\|\boldsymbol{Z}\|_0 \tag{3}$$

where, a transform $\boldsymbol{T}$ is learnt such that the computed coefficients $\boldsymbol{Z}$ are sparse. The second term in (3) is a regularizer where, the factor $-\log\det \boldsymbol{T}$ imposes a full rank on the learned transform and $\|\boldsymbol{T}\|_F^2$ balances the scale. This term is added to prevent trivial solutions and control the condition number of $\boldsymbol{T}$. Solving the optimization problem in (3) using alternate minimization technique results in the following closed form updates for $\boldsymbol{T}$ and $\boldsymbol{Z}$ [19]:

$$\boldsymbol{Z} \leftarrow \min_{\boldsymbol{Z}} \|\boldsymbol{TX} - \boldsymbol{Z}\|_F^2 + \mu\|\boldsymbol{Z}\|_0 \tag{4}$$

$$\boldsymbol{Z} = (abs(\boldsymbol{TX}) \geq \mu).\ \boldsymbol{TX} \tag{5}$$

where, the term in the bracket is hard thresholded against the value μ and '.' denotes the element-wise product.

$$\boldsymbol{T} \leftarrow \min_{\boldsymbol{T}} \|\boldsymbol{TX} - \boldsymbol{Z}\|_F^2 + \lambda(\|\boldsymbol{T}\|_F^2 - \log\det \boldsymbol{T}). \tag{6}$$

Using Cholesky decomposition followed by singular value decomposition we get:

$$\boldsymbol{X}\boldsymbol{X}^T + \lambda \boldsymbol{I} = \boldsymbol{L}\boldsymbol{L}^T \tag{7}$$

$$\boldsymbol{L}^{-1}\boldsymbol{X}\boldsymbol{Z}^T = \boldsymbol{U}\boldsymbol{S}\boldsymbol{V}^T \tag{8}$$

$$\hat{\boldsymbol{T}} = 0.5\boldsymbol{V}(\boldsymbol{S} + (\boldsymbol{S}^2 + 2\lambda \boldsymbol{I})^{1/2})\boldsymbol{U}^T\boldsymbol{L}^{-1}. \tag{9}$$

This is guaranteed to be a global optimum compared to iterative optimization methods such as conjugate gradients and it allows for both cheap and exact computation of the transform [6]. This framework can be used to learn the data representation and carry out classification or regression tasks depending whether the output variable is discrete or continuous.

2.2 Kernel Transform Learning

To capture the non-linearities in the data, the kernel transform learning problem is formulated as [19]:

$$\boldsymbol{B}\boldsymbol{\mathcal{K}}(\boldsymbol{X}, \boldsymbol{X}) = \boldsymbol{Z} \tag{10}$$

where $\boldsymbol{B}$ is the transform and $\boldsymbol{\mathcal{K}}(\boldsymbol{X}, \boldsymbol{X})$ is the kernel matrix which can be defined upfront unlike in dictionary based methods and is expressed as:

$$\boldsymbol{\mathcal{K}}(\boldsymbol{X}, \boldsymbol{X}) = \boldsymbol{\varphi}(\boldsymbol{X})^T\boldsymbol{\varphi}(\boldsymbol{X}) \tag{11}$$

$$\boldsymbol{B}\boldsymbol{\varphi}(\boldsymbol{X})^T\boldsymbol{\varphi}(\boldsymbol{X}) = \boldsymbol{Z}. \tag{12}$$

The overall formulation of kernel transform learning involving sparsity is given as:

$$\min_{\boldsymbol{B},\boldsymbol{Z}} \|\boldsymbol{B}\boldsymbol{\mathcal{K}}(\boldsymbol{X}, \boldsymbol{X}) - \boldsymbol{Z}\|_F^2 + \lambda(\|\boldsymbol{B}\|_F^2 - \log\det \boldsymbol{B}) + \mu\|\boldsymbol{Z}\|_0. \tag{13}$$

The closed form solution of transform and coefficients in the kernel transform learning remain the same as in the case of basic transform learning with the difference that in the former case, the kernelized version of input data is utilized instead of raw input data.

3 Proposed Transform Learning Methodology for Regression

With the brief introduction to transform learning in the previous section, the proposed formulation for regression considering both the basic and kernel versions of transforms is presented in the following.

3.1 Transform Learning for Regression (TLR)

Given a multi-variate data of N samples, let $\boldsymbol{X} \in \mathbb{R}^{L \times N}$ denote the feature vector of length L (i.e. independent variables of the regressor) and $\boldsymbol{y} \in \mathbb{R}^{1 \times N}$ represent the output of the regressor (i.e. dependent variable(s)). The transform learning framework can be utilized for regression tasks by adding a ridge regression penalty term and carrying out a joint optimization to learn the transform, coefficients and regression weights together. The said joint optimization problem is given as:

$$\min_{\boldsymbol{T},\boldsymbol{Z},\boldsymbol{w}} \|\boldsymbol{T}\boldsymbol{X} - \boldsymbol{Z}\|_F^2 + \lambda(\|\boldsymbol{T}\|_F^2 - \log\det \boldsymbol{T}) + \gamma\|\boldsymbol{y} - \boldsymbol{w}\boldsymbol{Z}\|_2^2 \tag{14}$$

where, $\boldsymbol{w} \in \mathbb{R}^{1 \times K}$ are the regression weights with K being the number of atoms in the transform $\boldsymbol{T}$. Here, sparsity constraint on $\boldsymbol{Z}$ is not considered because the transform learnt to model the data is not overcomplete.

The transform, coefficients and the regression weights are learnt in the ***training phase*** and later utilized in the test phase for regression tasks. Alternate minimization technique is used to compute the closed form updates for all the variables. The sub-problems to solve in the training phase are:

$$\boldsymbol{Z} \leftarrow \min_{\boldsymbol{Z}} \underbrace{\|\boldsymbol{T}\boldsymbol{X} - \boldsymbol{Z}\|_F^2 + \gamma\|\boldsymbol{y} - \boldsymbol{w}\boldsymbol{Z}\|_2^2}_{M} \tag{15}$$

$$\boldsymbol{w} \leftarrow \min_{\boldsymbol{w}} \gamma\|\boldsymbol{y} - \boldsymbol{w}\boldsymbol{Z}\|_2^2. \tag{16}$$

For the transform update $\boldsymbol{T}$, the problem remains the same as in (6) and hence the closed form solution is similar to (9). The closed form update for $\boldsymbol{w}$ is direct and computed as $\boldsymbol{w} = \boldsymbol{y}\boldsymbol{Z}^{\dagger}$, where '†' denotes pseudo inverse. Solving for $\boldsymbol{Z}$, for any matrix $\boldsymbol{A}$, using the identity $\|\boldsymbol{A}\|_F^2 = \mathrm{tr}(\boldsymbol{A}^T\boldsymbol{A})$ and using the first order derivative condition $\frac{\partial M}{\partial Z} = 0$, we get the following closed form update:

$$\boldsymbol{Z} = (\boldsymbol{1} + \gamma\boldsymbol{w}^T\boldsymbol{w})^{\dagger}(\boldsymbol{T}\boldsymbol{X} + \gamma\boldsymbol{w}^T\boldsymbol{y}). \tag{17}$$

In the ***test phase***, given new test samples $\boldsymbol{x_{test}} \in \mathbb{R}^{L \times n}$, the dependent variable $\hat{\boldsymbol{y}}_{\boldsymbol{test}}$ is computed using the model learnt in terms of the $\boldsymbol{T}$ and $\boldsymbol{w}$ in the training phase. The coefficients $\boldsymbol{z_{test}}$ and output $\hat{\boldsymbol{y}}_{\boldsymbol{test}}$ are computed as:

$$\boldsymbol{z_{test}} = \boldsymbol{T}\boldsymbol{x_{test}} \tag{18}$$

$$\hat{\boldsymbol{y}}_{\boldsymbol{test}} = \boldsymbol{w}\boldsymbol{z_{test}}. \tag{19}$$

This formulation can be extended to kernel transform learning based methods, which effectively capture the complex non-linear relationship in the data and is described in the next section.

3.2 Kernel Transform Learning for Regression (KTLR)

Similar to transform learning based method, kernel transform learning method can also be formulated to cater for regression problems by inserting an additional ridge regression penalty term. The regression formulation is given as:

$$\min_{\boldsymbol{B},\boldsymbol{Z},\boldsymbol{w}} \|\boldsymbol{B}\mathcal{K}(\boldsymbol{X},\boldsymbol{X}) - \boldsymbol{Z}\|_F^2 + \lambda(\|\boldsymbol{B}\|_F^2 - \log\det\boldsymbol{B}) + \gamma\|\boldsymbol{y} - \boldsymbol{w}\boldsymbol{Z}\|_2^2. \tag{20}$$

In the ***training phase***, since the update of transform $\boldsymbol{B}$ is not affected by the added ridge regression penalty term, it remains the same as given for $\boldsymbol{T}$ (*i.e.* from (7) to (9)) with the only difference that the input data $\boldsymbol{X}$ in (7) is replaced by the kernelized version of the data $\mathcal{K}(\boldsymbol{X},\boldsymbol{X})$.

To solve for $\boldsymbol{Z}$, the following sub-problem is required to be solved:

$$\boldsymbol{Z} \leftarrow \min_{\boldsymbol{Z}} \|\boldsymbol{B}\mathcal{K}(\boldsymbol{X},\boldsymbol{X}) - \boldsymbol{Z}\|_F^2 + \gamma\|\boldsymbol{y} - \boldsymbol{w}\boldsymbol{Z}\|_2^2. \tag{21}$$

The closed form update for $\boldsymbol{Z}$ is similar to (17) and can be expressed as:

$$\boldsymbol{Z} = (\boldsymbol{1} + \gamma\boldsymbol{w}^T\boldsymbol{w})^{\dagger}(\boldsymbol{B}\mathcal{K}(\boldsymbol{X},\boldsymbol{X}) + \gamma\boldsymbol{w}^T\boldsymbol{y}). \tag{22}$$

The update of $\boldsymbol{w}$ is identical to the previous case.

In the ***test phase***, the coefficient $\boldsymbol{z_{test}}$ is computed utilizing the transform $\boldsymbol{B}$ learnt in the training phase and is given by:

$$\boldsymbol{B}\boldsymbol{\varphi}(\boldsymbol{x_{test}})^T\boldsymbol{\varphi}(\boldsymbol{X}) = \boldsymbol{z_{test}}. \tag{23}$$

The output $\hat{\boldsymbol{y}}_{\boldsymbol{test}}$ is computed in the same way as in (19).

The pseudo code of TLR and KTLR algorithm is summarized in Algorithm 1.

4 Experimental Results and Discussion

In this section, we demonstrate the functionality of the proposed framework of transform learning for regression tasks. Apart from synthetic data, two real-life datasets are considered for performance evaluation. For comparative study, the estimation results of the proposed TLR and KTLR algorithms are presented along with those obtained from Kernel Regression (KR) [21], Linear Regression (LR), Dictionary counterparts for regression (Kernel Dictionary Learning for Regression (KDLR), Dictionary Learning for Regression (DLR)). The dictionary (DLR and KDLR) methods consider similar joint optimization mentioned in (14,

Algorithm 1 Transform and Kernel Transform Learning for Regression (TLR or KTLR)

Input: Set of training data, $\boldsymbol{X} = \boldsymbol{X_{train}}$, $\boldsymbol{y} = \boldsymbol{y_{train}}$, K (size of transform (atoms)), parameters (λ, γ) and kernel function $\boldsymbol{\kappa}$ to compute kernel matrix $\boldsymbol{\mathcal{K}(X, X)}$, $\boldsymbol{\mathcal{K}(x_{test}, X)}$ and test data $\boldsymbol{x_{test}}$
Output: Learnt transform $\boldsymbol{T}$ or $\boldsymbol{B}$, weight vector $\boldsymbol{w}$, estimated output $\boldsymbol{\hat{y}_{test}}$
Initialization: Set $\boldsymbol{Z}_0$ to random matrix with real numbers between 0 and 1 drawn from a uniform distribution, $\boldsymbol{w}_0 = yZ^{\dagger}$ and $\boldsymbol{T}_0$ or $\boldsymbol{B}_0 = \boldsymbol{O}$, iteration $i = 1$

procedure
  loop: Repeat until convergence (or fixed number of iterations $Maxitr$)
  $\boldsymbol{T}_i$ or $\boldsymbol{B}_i \leftarrow 0.5\boldsymbol{V}_i(\boldsymbol{S}_i + (\boldsymbol{S}_i^2 + 2\lambda \boldsymbol{I})^{1/2})\boldsymbol{U}_i^T \boldsymbol{L}_i^{-1}$
  $\boldsymbol{Z}_i \leftarrow$ update using $\boldsymbol{T}_i$ or $\boldsymbol{B}_i$ & $\boldsymbol{w}_{i-1}$ using (17 or 22)
  $\boldsymbol{w}_i \leftarrow \boldsymbol{y}\boldsymbol{Z}_i^{\dagger}$
  $i \leftarrow i + 1$
  if $\|\boldsymbol{T}_i(or \boldsymbol{B}_i) - \boldsymbol{T}_{i-1}(or \boldsymbol{B}_{i-1})\|_F < Tol$ or $i == Maxitr$ **then**
    $\boldsymbol{z_{test}} \leftarrow$ update using (18 or 23)
    $\boldsymbol{\hat{y}_{test}} \leftarrow \boldsymbol{w z_{test}}$
    close;
  else go to *loop*

20) for learning the dictionary and regression weights together [15]. For all the datasets, the data is normalized (using min-max normalization) before subjecting it to different regression methods and the estimation results are appropriately scaled back. The kernel based methods make use of the 'radial basis function' kernel for fair comparison with the scaling factor $\sigma = 0.8$. Three metrics namely, Mean Squared Error (MSE), Mean Absolute Error (MAE) and Pearson's Correlation Coefficient (PCC) are used for performance evaluation of the algorithms.

A. Synthetic Dataset: The simulated non-linear data considered in [22], [23] is used for evaluation by taking 3 predictors and 1 response variable of the data of length 500 samples. Randomly 90% of the data ($N = 450$) is used for training and the remaining for testing. The estimation results of the response variable with different methods are presented in Fig 1. Table I summarizes the performance metrics obtained with all the methods using 5-fold cross-validation. It can be seen, for the linear case, TLR, DLR and LR all have similar performance. For the kernel case, KTLR has the least MSE and performs slightly better than KR.

B. Public Dataset: One of the UCI datasets, *Energy Efficiency* is considered for regression analysis. This dataset is used to assess the heating and cooling load requirements of buildings as a function of 8 building parameters (Relative compactness, Surface area, Wall area, Roof area, Overall height, Orientation, Glazing area, Glazing area distribution) [24]. The dataset comprises of 768 samples from 12 different building shapes. Here again, randomly 90% of the data ($N = 691$) is considered for training and the remaining for testing. It is worth noting that, although the proposed formulation considers single output variables, the dictionary and transform based techniques discussed here can be

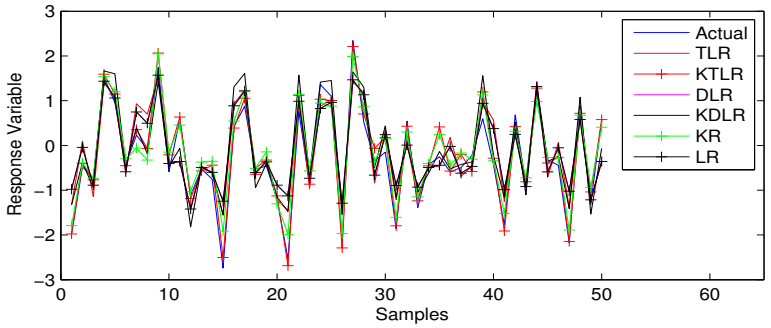

Fig. 1. Response Variable Estimation

Table 1. Results with Synthetic Dataset

Algorithm	MSE	MAE	PCC
KTLR	**0.0017 ± 0.0003**	**0.0374 ± 0.0022**	**0.963**
TLR	0.0088 ± 0.0017	0.0755 ± 0.0079	0.836
KDLR	0.0047 ± 0.0016	0.0510 ± 0.0088	0.924
DLR	0.0087 ± 0.0015	0.0750 ± 0.0062	0.837
KR	0.0024 ± 0.0005	0.0413 ± 0.0043	0.963
LR	0.0087 ± 0.0015	0.0750 ± 0.0062	0.837

extended to handle multiple output variables in the same way. Here, a joint set of weights are computed for the dual outputs (i.e. heating and cooling loads). The estimation results of the heating load with different methods are presented in Fig 2. Tables II and III give the summary of the results obtained for heating and cooling load estimation respectively using 5-fold cross-validation. Similar to the previous case, KTLR performs better than other kernel methods. It can be seen, for heating load estimation, KTLR has the least MSE and highest PCC and for the cooling load estimation, its performance is similar to KR.

Table 2. Results for Heating Load Estimation

Algorithm	MSE	MAE	PCC
KTLR	**0.0045 ± 0.0011**	**0.0497 ± 0.0080**	**0.974**
TLR	0.0077 ± 0.0016	0.0693 ± 0.0068	0.953
KDLR	0.0065 ± 0.0034	0.0601 ± 0.0118	0.961
DLR	0.0095 ± 0.0012	0.0778 ± 0.0050	0.942
KR	0.0053 ± 0.0009	0.0521 ± 0.0056	0.973
LR	0.0076 ± 0.0016	0.0645 ± 0.0061	0.955

C. Building Power Consumption Data: Power consumption data from office building is considered for electrical load forecasting. Load forecasting is

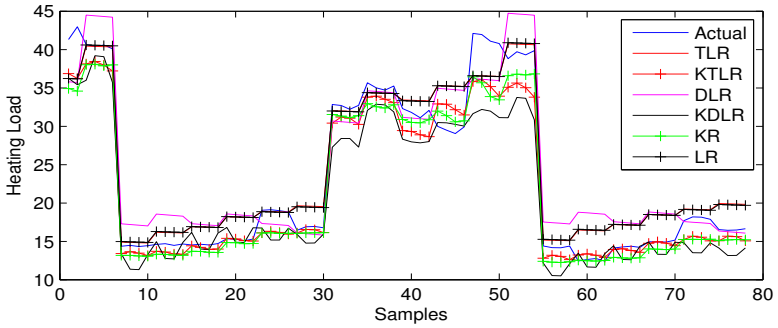

Fig. 2. Heating Load Estimation for Building

Table 3. Results for Cooling Load Estimation

Algorithm	**MSE**	**MAE**	**PCC**
KTLR	**0.0070 ± 0.0009**	**0.0585 ± 0.0047**	**0.958**
TLR	0.0098 ± 0.0027	0.0690 ± 0.0123	0.938
KDLR	0.0085 ± 0.0045	0.0610 ± 0.0108	0.940
DLR	0.0100 ± 0.0080	0.0705 ± 0.0201	0.923
KR	**0.0070 ± 0.0022**	**0.0578 ± 0.0087**	**0.961**
LR	0.0098 ± 0.0039	0.0692 ± 0.0126	0.938

framed as a function approximation problem where, temperature, previous day power consumption, previous week same day power consumption data and contextual information like weekdays and weekends are considered as potential inputs for producing a day ahead power consumption forecast. Office building data comprises of aggregate power consumption data for 6 months, sampled every 15 minutes. To keep track of seasonal changes, one month data ($N = 2880$) is used for training to produce day ahead forecast ($n = 96$). Although it could be extended to produce week ahead forecast in the same way, results with day ahead forecast alone is presented to demonstrate the applicability of the proposed method for time-series forecasting. The day ahead forecast results with KTLR, TLR and KR are presented in Fig 3. Table IV gives the estimation results obtained with different methods for day ahead forecast averaged over 7 days. As is evident from the plot, KTLR is able to track the actual load much better than KR and TLR. Figure 4 presents the boxplot of the MAE which demonstrate the superior performance of KTLR compared to other methods.

It is worth noting that the performance of dictionary and transform-based regression depends on the number of atoms and how well the hyperparameters of the optimization function are tuned. In this work, the hyperparameters for these methods are obtained through extensive search and tuned appropriately for each dataset. The results presented here made use of the best configurations (in terms of atoms size K) of the respective dictionaries and transforms for making fair comparisons.

Table 4. Results with Building Power Consumption Data

Algorithm	MSE	MAE	PCC
KTLR	**0.0016 ± 0.0022**	**0.0218 ± 0.0082**	**0.976**
TLR	0.0046 ± 0.0029	0.0529 ± 0.0148	0.946
KR	0.0018 ± 0.0021	0.0252 ± 0.0078	0.971

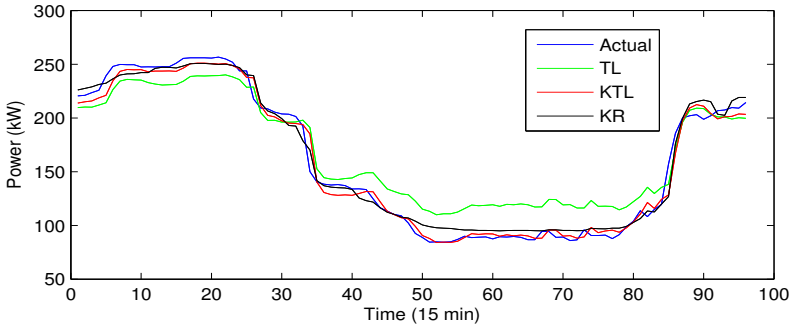

Fig. 3. Day ahead Load Forecast for Building

5 Conclusion

The paper systematically presented the transform learning based approach for regression using a joint optimization of the transform, associated coefficients and the regression weights. The results obtained with different datasets are presented which demonstrate the potential and utility of the proposed function approximation technique for modeling and forecasting of the time series (signals).

References

1. Bruno A. Olshausen and David J. Field, "Sparse coding with an overcomplete basis set: A strategy employed by v1?," *Vision Research*, vol. 37, no. 23, pp. 3311 – 3325, 1997.
2. M. Aharon, M. Elad, and A. Bruckstein, "K-svd: An algorithm for designing overcomplete dictionaries for sparse representation," *IEEE Transactions on Signal Processing*, vol. 54, no. 11, pp. 4311–4322, Nov 2006.
3. Julien Mairal, Jean Ponce, Guillermo Sapiro, Andrew Zisserman, and Francis R. Bach, "Supervised dictionary learning," in *Advances in Neural Information Processing Systems 21*, D. Koller, D. Schuurmans, Y. Bengio, and L. Bottou, Eds., pp. 1033–1040. Curran Associates, Inc., 2009.
4. G. Chen and D. Needell, "Compressed sensing and dictionary learning," *Finite Frame Theory: A Complete Introduction to Overcompleteness*, vol. 73, 2016.
5. S. Ravishankar and Y. Bresler, "Learning sparsifying transforms," *IEEE Transactions on Signal Processing*, vol. 61, no. 5, pp. 1072–1086, March 2013.
6. S. Ravishankar, B. Wen, and Y. Bresler, "Online sparsifying transform learning part i: Algorithms," *IEEE Journal of Selected Topics in Signal Processing*, vol. 9, no. 4, pp. 625–636, June 2015.

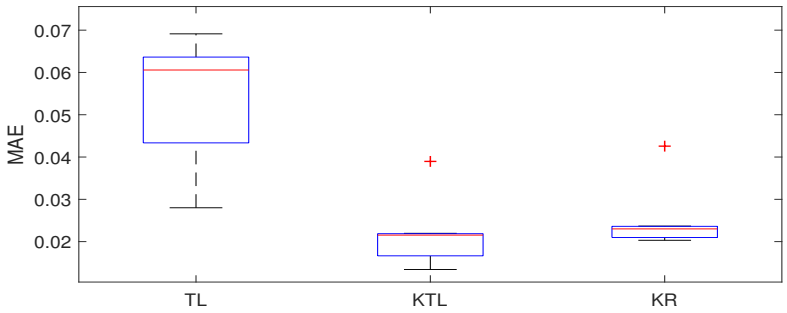

Fig. 4. Box Plot of MAE for Day ahead Load Forecast

7. S. Ravishankar and Y. Bresler, "Learning overcomplete sparsifying transforms for signal processing," in *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*, May 2013, pp. 3088–3092.
8. S. Ravishankar and Y. Bresler, "Efficient blind compressed sensing using sparsifying transforms with convergence guarantees and application to magnetic resonance imaging," *SIAM Journal on Imaging Sciences*, vol. 8, no. 4, pp. 2519–2557, 2015.
9. Ivana Tosic and Pascal Frossard, "Dictionary learning: What is the right representation for my signal?," *IEEE Signal Processing Magazine*, vol. 28, no. 2, pp. 27–38, 2011.
10. Y. Bengio, A. Courville, and P. Vincent, "Representation learning: A review and new perspectives," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 8, pp. 1798–1828, Aug 2013.
11. W. Tang, I. R. Otero, H. Krim, and L. Dai, "Analysis dictionary learning for scene classification," in *2016 IEEE Statistical Signal Processing Workshop (SSP)*, June 2016, pp. 1–5.
12. Jun Guo, Yanqing Guo, Xiangwei Kong, Man Zhang, and Ran He, "Discriminative analysis dictionary learning," in *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*. 2016, AAAI'16, pp. 1617–1623, AAAI Press.
13. H. V. Nguyen, V. M. Patel, N. M. Nasrabadi, and R. Chellappa, "Kernel dictionary learning," in *2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, March 2012, pp. 2021–2024.
14. R. Ganti and R. M. Willett, "Sparse Linear Regression With Missing Data," *ArXiv e-prints*, Mar. 2015.
15. K. Kumar, A. Majumdar, M. G. Chandra, and A. Anil Kumar, "Regressing kernel dictionary learning," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, April 2018.
16. Chengyu Peng, Hong Cheng, and Manchor Ko, "An efficient two-stage sparse representation method," *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 30, no. 01, pp. 1651001, 2016.
17. Z. Jiang, Z. Lin, and L. S. Davis, "Label consistent k-svd: Learning a discriminative dictionary for recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 11, pp. 2651–2664, Nov 2013.
18. S. Ravishankar and Y. Bresler, "Online sparsifying transform learning part ii: Convergence analysis," *IEEE Journal of Selected Topics in Signal Processing*, vol. 9, no. 4, pp. 637–646, June 2015.

19. J. Maggu and A. Majumdar, "Kernel transform learning," *Pattern Recognition Letters*, vol. 98, pp. 117 – 122, 2017.
20. S. Ravishankar and Y. Bresler, "Learning doubly sparse transforms for images," *IEEE Transactions on Image Processing*, vol. 22, no. 12, pp. 4598–4612, Dec 2013.
21. Yi Cao, "Multivariant Kernel Regression and Smoothing," `https://in.mathworks.com/matlabcentral/fileexchange/19279`, March 2008.
22. W. W. Hsieh, "Nonlinear canonical correlation analysis by neural networks," *Neural Netw.*, vol. 13, no. 10, pp. 1095–1105, Dec. 2000.
23. William W. Hsieh, *Machine Learning Methods in the Environmental Sciences: Neural Networks and Kernels*, Cambridge University Press, New York, NY, USA, 1st edition, 2009.
24. A. Xifara A. Tsanas, "Accurate quantitative estimation of energy performance of residential buildings using statistical machine learning tools," in *Energy and Buildings*, 2012, vol. 49, pp. 560–567.