Graph Structured Dictionary for Regression

Kriti Kumar, M Girish Chandra, Aakanksha Bapna and A Anil Kumar
TCS Research and Innovation, Bangalore, India.
Email: {kriti.kumar, m.gchandra, aakanksha.bapna, achannaanil.kumar}@tcs.com

***Abstract*—Transformations for signals defined on graphs are playing significant role in applying the emerging graph signal processing techniques to different tasks. In this paper we focus on utilizing graph signal dictionary, a data-driven transformation, for regression. Apart from spelling out the joint optimization formulation, as well as the associated iteration steps to arrive at the dictionary and the regression coefficients, the paper provides some initial results to bring out the usefulness of the proposed approach.**

I. Introduction

Representation Learning (RL) refers to learning representations of the data that make it easier to extract useful information when building classifiers or other predictors (like regressors) [1]. One instance of RL being dictionary learning, which lead to a compact representation of the data in many situations. Traditionally, they have been used extensively for analysis and synthesis problems, especially arising in image processing [2], [3], [4], [5]. Many natural signals have sparse representation when appropriate dictionary is used for decomposition. The basic formulation for dictionary learning is given as:

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

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

Dictionary learning originated from matrix factorization [6] and sparse coding [7] problems. Method of Optimal Directions (MOD) [8] is used to solve the standard matrix factorization problem by alternately solving for the two variables $\boldsymbol{D}$ and $\boldsymbol{Z}$:

$$\min_{\boldsymbol{D},\boldsymbol{Z}} \|\boldsymbol{X} - \boldsymbol{D}\boldsymbol{Z}\|_F^2 \tag{2}$$

On the other hand, K-SVD algorithm [9] is more popular for solving sparse coding problems, where $\boldsymbol{Z}$ is constrained to be sparse. This leads to the following formulation:

$$\min_{\boldsymbol{D},\boldsymbol{Z}} \|\boldsymbol{X} - \boldsymbol{D}\boldsymbol{Z}\|_F^2 \, s.t. \|\boldsymbol{Z}\|_0 \leq \boldsymbol{\tau} \tag{3}$$

where, $\|\boldsymbol{Z}\|_0$ is the l_0 sparsity measure which ensures that the optimization results in a sparse representation of data learnt using at most $\boldsymbol{\tau}$ non-zero entries of $\boldsymbol{Z}$.

Dictionary Learning based techniques have been also explored for classification tasks [10], [11]. There are both two-stage approaches [12], where the sparse coefficients and dictionary are learnt separately, keeping the other fixed and single-stage approach [13], where a joint optimization framework is used to learn the dictionary, coefficient and classification weights together. Few works also report the use of sparse coefficients as features which are fed to conventional machine learning models to carry out classification [7], [14]. The application of dictionary towards regression tasks is, however limited [15], [16]. The work in [15], presents a fast method for sparse regression in the presence of missing data. The other work [16] proposed a kernel dictionary learning based framework for regression which could capture the non-linearities in the data and outperform the conventional Kernel Regression.

With the signals exhibiting a graph nature in applications like, social, economic, biological networks, transportation and sensor networks, there is a need to analyze this data for both classification and regression tasks. Some works [17], [18] have been reported on representation learning using dictionary which utilizes the underlying structure of the data. Few attempts have been made [19], [20], where the smoothness on the manifold graph Laplacian is used as an additional constraint for learning the dictionary and sparse coefficients for classification tasks. The smoothness in the Graph Signal Processing (GSP) domain is described in terms of Graph Total Variation (GTV) given as:

$$GTV = \boldsymbol{s}^T\boldsymbol{L}\boldsymbol{s} = \sum_{(i,j)\in E} \boldsymbol{W}(i,j)[s(i) - s(j)]^2 \tag{4}$$

where $\boldsymbol{W}(i,j)$ is the weight assigned to the connection between vertices (i,j), $\boldsymbol{s}$ is the graph signal and $\boldsymbol{L}$ is the graph Laplacian. The $\boldsymbol{W}(i,j)$ representing the graph is constructed such that similar signal values at the vertices are connected with high weights and vice-versa. The smoothness expression when evaluated, eventually results in small value of graph total variation, suggesting the signal is smooth with respect to the underlying graph. Using this notion of smoothness, the manifold regularized sparse coding is formulated as [21]:

$$\min_{\boldsymbol{D},\boldsymbol{Z}} \|\boldsymbol{X} - \boldsymbol{D}\boldsymbol{Z}\|_F^2 + \beta Trace(\boldsymbol{Z}\boldsymbol{L}\boldsymbol{Z}^{\boldsymbol{T}}) \, s.t. \|\boldsymbol{Z}\|_0 \leq \boldsymbol{\tau} \tag{5}$$

where, $\boldsymbol{L}$ here is the manifold Laplacian, which represent the correlation between data elements. It is worth noting that, the important notion of graph smoothness which is extensively used in GSP formulations, is captured in the second term involving the trace. The work in [20] learns the graph Laplacian in the joint optimization framework along with the sparse coefficients and dictionary atoms such that the manifold geometry is preserved. Coming to the popular Machine Learning tasks, so far, only classification problems have been addressed using the graph structured dictionary.

Extending the work to regression tasks, using dictionary learnt for signals defined over graphs, in this paper, Graph

ISBN 978-90-827970-1-5

Structured Dictionary Learning framework for Regression (GDLR) is proposed. To the best of our knowledge, there has been no prior study on graph structured dictionary learning where the regression weights are learnt within the dictionary learning framework. The proposed method exploits the underlying structure of the data while learning the dictionary and weights of the regression model in a joint optimization framework. With the underlying structure of the data captured appropriately by the graph, this technique outperforms the traditional Linear Regression (LR), Kernel Regression (KR) and Dictionary Learning for Regression (DLR).

Towards providing the formulation of the proposed methodology and demonstrating its potential for regression tasks, this paper is organized as follows. Section II gives a brief background on graph and graph signals. The detailed formulation of the proposed Graph structured Dictionary Learning framework for Regression (GDLR) is given in section III. Section IV presents the results of the proposed method obtained using synthetic and real-life dataset. This is followed by section V which concludes the work.

II. Brief Background on Graph and Graph Signals

Graphs are generic models which are used to represent the complex relationship between data elements of any arbitrary dataset. A graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ is represented by a set of vertices $\mathcal{V}$, where $\mathcal{V} = \{v_n\}_{n=1}^{N}$ and a set of edges $\mathcal{E}$ connecting the vertices. The adjacency matrix $\boldsymbol{A} \in \mathbb{C}^{N\times N}$ is used to define the connections between vertices using edges. If there exists an edge $e = (i, j)$ between vertices i and j, $\boldsymbol{A}(i, j) = 1$ otherwise it is 0. The weight matrix $\boldsymbol{W} \in \mathbb{C}^{N\times N}$ assigns weights to the connections in $\boldsymbol{A}$ by capturing the complex interactions between the vertices. Usually these two matrices are defined by the application. Otherwise, one way is to estimate it directly from the data [22], or use a thresholded Gaussian kernel weighting function [23], [24] expressed as:

$$\boldsymbol{W}(i,j) = \begin{cases} exp(-|dist(i,j)|^2/\sigma^2), & \text{if } dist(i,j) \leq \kappa. \\ 0, & \text{otherwise.} \end{cases} \quad (6)$$

where, σ is a scaling factor and $dist(i, j)$ is the distance measure between vertices i and j which can be either the Physical distance, Euclidean distance or Cosine distance etc. κ is the threshold applied on the distance measure which decides the connection between the vertices.

Another important matrix associated with graphs is the graph Laplacian $\boldsymbol{L}$ which contains most of the information of the graph and its properties [24]. It is a real symmetric and positive semi-definite matrix. For an undirected graph, the unnormalized graph Laplacian $\boldsymbol{L}$ is defined as: $\boldsymbol{L} = \boldsymbol{D} - \boldsymbol{W}$. where, $\boldsymbol{W}$ is the weight matrix and $\boldsymbol{D}$ is the degree matrix, which is a diagonal matrix containing the sum of weighted edges incident on the vertices of the graph along its diagonal, and is expressed as: $\boldsymbol{D}(i, i) = \sum_j \boldsymbol{W}(i, j)$. Normalized graph Laplacian $\boldsymbol{L_{norm}}$, is obtained as:

$$\boldsymbol{L_{norm}} = \boldsymbol{D}^{-\frac{1}{2}}\boldsymbol{L}\boldsymbol{D}^{-\frac{1}{2}} \quad (7)$$

The signal on the graph or graph signal $\boldsymbol{s}$ is a signal whose samples reside on the vertices of the graph. Graph signal as defined in [24] is a signal or function $\boldsymbol{s} : \mathcal{V} \to \mathbb{R}$ defined on the vertices of the graph may be represented as a vector $\boldsymbol{s} \in \mathbb{R}^N$, where the ith component of the vector $\boldsymbol{s}$ represents the function value at the ith vertex in $\mathcal{V}$. The emerging field of GSP provides signal processing tools which work on signals having an underlying graph structure [24]. An important tool used for analysing these signals is Graph Fourier Transform (GFT). It is computed as: $\hat{\boldsymbol{s}} = \boldsymbol{U}^{-1}\boldsymbol{s}$, where $\boldsymbol{U}$ are the eigenvectors obtained by the eigen-decomposition of either $\boldsymbol{L}$ or $\boldsymbol{W}$. The Inverse Graph Fourier Transform (IGFT) is given as: $\boldsymbol{s} = \boldsymbol{U}\hat{\boldsymbol{s}}$.

These concepts are used to address the problem of regression where the data elements exhibit a graph structure. The intrinsic structure of the signals is exploited for enhanced representation learning using a dictionary which is described next.

III. Graph Structured Dictionary Learning for Regression

Graph structured dictionary learning framework incorporates a ridge regression penalty and additional Laplacian regularization term in the objective function that enforces the coefficients to be smooth with respect to the intrinsic structure of the data. Given the manifold Laplacian, the dictionary, coefficients and regression weights are learnt together in a joint optimization framework. For a multivariate data of N training samples, mathematically, the proposed dictionary-based regression is formulated as:

$$\min_{\boldsymbol{D},\boldsymbol{Z},\boldsymbol{w}} \|\boldsymbol{X} - \boldsymbol{D}\boldsymbol{Z}\|_F^2 + \alpha\|\boldsymbol{y} - \boldsymbol{w}\boldsymbol{Z}\|_2^2 + \beta Trace(\boldsymbol{Z}\boldsymbol{L}\boldsymbol{Z}^{\boldsymbol{T}}) \quad (8)$$

where, $\boldsymbol{X} \in \mathbb{R}^{L\times N}$ represent the independent variables of feature vector length L and $\boldsymbol{y} \in \mathbb{R}^N$ represent the dependent variable. $\boldsymbol{D} \in \mathbb{R}^{L\times K}$ is the dictionary, containing atoms as its columns and $\boldsymbol{Z} \in \mathbb{R}^{K\times N}$ the learnt coefficients; $\boldsymbol{L} \in \mathbb{R}^{N\times N}$ is the manifold Laplacian of the data and $\boldsymbol{w} \in \mathbb{R}^K$ are the regression weights.

The sparsity term is not included in (8), since the focus is on regression, where undercomplete dictionaries would be used. Two phases are involved in carrying out regression using the proposed technique: *(i) Training phase*, where the dictionary and regression weights are learnt utilizing the graph structure, and *(ii) Test phase* where these learnt parameters are used for estimating the target or dependent variable. The requisite optimization problems to be solved in these two phases and their closed form solution are detailed below.

A. Training Phase

The standard alternating minimization approach is used to solve (8). The sub-problems required to be solved are:

$$\boldsymbol{D} \leftarrow \min_{\boldsymbol{D}} \|\boldsymbol{X} - \boldsymbol{D}\boldsymbol{Z}\|_F^2 \quad (9)$$

$$\boldsymbol{Z} \leftarrow \min_{\boldsymbol{Z}} \|\boldsymbol{X} - \boldsymbol{D}\boldsymbol{Z}\|_F^2 + \alpha\|\boldsymbol{y} - \boldsymbol{w}\boldsymbol{Z}\|_2^2 + \beta Trace(\boldsymbol{Z}\boldsymbol{L}\boldsymbol{Z}^{\boldsymbol{T}}) \quad (10)$$

ISBN 978-90-827970-1-5 © EURASIP 2018

$$\boldsymbol{w} \leftarrow \min_{\boldsymbol{w}} \|\boldsymbol{y} - \boldsymbol{w}\boldsymbol{Z}\|_2^2 \tag{11}$$

Each of the sub-problems have a closed form update; (9) and (11) are straightforward linear inverse problems; the closed form inverse is the Moore-Penrose pseudo-inverse. The update for $\boldsymbol{Z}$ is obtained by taking a derivative of (10) with respect to $\boldsymbol{Z}$ and equating it to 0. It results in a Sylvester equation of the form:

$$(\boldsymbol{D}^T\boldsymbol{D} + \boldsymbol{\alpha}\boldsymbol{w}^T\boldsymbol{w})\boldsymbol{Z} + \boldsymbol{Z}\boldsymbol{\beta}\boldsymbol{L} = \boldsymbol{D}^T\boldsymbol{X} + \boldsymbol{\alpha}\boldsymbol{w}^T\boldsymbol{y} \tag{12}$$

Since the eigenvalues of $(\boldsymbol{D}^T\boldsymbol{D} + \boldsymbol{\alpha}\boldsymbol{w}^T\boldsymbol{w})$ and $(-\boldsymbol{\beta}\boldsymbol{L})$ are distinct, a unique solution for $\boldsymbol{Z}$ exists [25].

B. Test Phase

Given a new test sample $\boldsymbol{x_{test}}$, the target or dependent variable $\hat{\boldsymbol{y}}_{\boldsymbol{test}}$ is estimated using the dictionary and regression weights learnt in the training phase. As a first step, the corresponding feature coefficients $\boldsymbol{z_{test}}$ are computed using the model expressed as:

$$\boldsymbol{x_{test}} = \boldsymbol{D}\boldsymbol{z_{test}} \tag{13}$$

Since the dictionary is known, the solution for $\boldsymbol{z_{test}}$ is formulated as:

$$\min_{\boldsymbol{z_{test}}} \|\boldsymbol{x_{test}} - \boldsymbol{D}\boldsymbol{z_{test}}\|_F^2 \tag{14}$$

The feature $\boldsymbol{z_{test}}$ once computed, is multiplied by the learnt regression weights to get the $\hat{\boldsymbol{y}}_{\boldsymbol{test}}$.

$$\hat{\boldsymbol{y}}_{\boldsymbol{test}} = \boldsymbol{w}\boldsymbol{z_{test}} \tag{15}$$

The pseudo code of the GDLR algorithm is presented in Algorithm 1.

Algorithm 1 Graph Structured Dictionary Learning for Regression (GDLR)

Input: Set of training data, $\boldsymbol{X} = \boldsymbol{X_{train}}$, $\boldsymbol{y} = \boldsymbol{y_{train}}$, $\boldsymbol{K}$ (size of dictionary), regularization parameters ($\boldsymbol{\alpha}$, $\boldsymbol{\beta}$), Graph Laplacian $\boldsymbol{L}$, test data $\boldsymbol{x_{test}}$

Output: Learnt dictionary $\boldsymbol{D}$, weight vector $\boldsymbol{w}$, estimated output $\hat{\boldsymbol{y}}_{\boldsymbol{test}}$

Initialization: Set $\boldsymbol{Z_0}$ to random matrix with real numbers between 0 and 1 drawn from a uniform distribution, $\boldsymbol{w_0} = \boldsymbol{y}\boldsymbol{Z}^\dagger$ († denotes pseudo inverse) and $\boldsymbol{D_0} = \boldsymbol{O}$, iteration $i = 1$

1: **procedure**
2: *loop*: Repeat until convergence (or fixed number of iterations $Maxitr$)
3: $\quad\boldsymbol{D_i} \leftarrow \boldsymbol{X}\boldsymbol{Z}_{\boldsymbol{i-1}}^\dagger$
4: $\quad$Normalize each column in $\boldsymbol{D_i}$ to a unit norm
5: $\quad\boldsymbol{Z_i} \leftarrow$ update using $\boldsymbol{D_i}$ and $\boldsymbol{w_{i-1}}$ using (10)
6: $\quad\boldsymbol{w_i} \leftarrow \boldsymbol{y}\boldsymbol{Z}_{\boldsymbol{i}}^\dagger$
7: $\quad i \leftarrow i + 1$
8: $\quad$**if** $\|\boldsymbol{D_i} - \boldsymbol{D_{i-1}}\|_F < Tol$ or $i == Maxitr$ **then**
9: $\quad\quad\boldsymbol{z_{test}} \leftarrow \boldsymbol{D}^\dagger\boldsymbol{x_{test}}$
10: $\quad\quad\hat{\boldsymbol{y}}_{\boldsymbol{test}} \leftarrow \boldsymbol{w}\boldsymbol{z_{test}}$
11: $\quad\quad$**close**;
12: $\quad$**else go to** *loop*

IV. Performance Study and Discussion

This section discusses the performance of the proposed graph structured dictionary framework for regression tasks. The results are demonstrated using both synthetic data and real-life data. For comparative study, the estimation results obtained with the proposed GDLR method are presented along with those obtained from traditional Linear Regression (LR), Kernel Regression (KR) [26] and traditional Dictionary Learning (DLR) framework. DLR method makes use of the similar joint optimization framework mentioned in (8) but, without the Laplacian regularization term. It is worth noting that the performance of any graph structured dictionary learning is sensitive to how well the graph Laplacian $\boldsymbol{L}$ is learnt from the data. So it is necessary that the learnt $\boldsymbol{L}$ is able to represent the relationship between data elements. This work makes use of the Gaussian kernel weighting function given in (6) to compute the graph structure from which $\boldsymbol{L}$ is estimated. Mean Squared Error (MSE) and Mean Absolute Error (MAE) are used as the performance metrics for evaluating different methods. The parameters $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ for GDLR and $\boldsymbol{\alpha}$ for DLR are carefully tuned after exhaustive grid search.

A. Synthetic Data

The performance of the proposed framework is assessed with synthetic data where the dictionary, regression weights and the underlying graph structure of the multi-variate data is known. A random sensor graph of $N = 100$ vertices is constructed using the GSP toolbox for Matlab [27] from which the graph Laplacian, $\boldsymbol{L_{gen}}$ is obtained. Five signals corresponding to multi-variate data are defined in the spectral domain $\boldsymbol{s}$ ($\boldsymbol{s_1}$ to $\boldsymbol{s_5}$), as heat kernels [27] with different values for τ (5, 3, 2, 1, 0.5). The corresponding signals in the vertex domain, $\boldsymbol{x}$ ($\boldsymbol{x_1}$ to $\boldsymbol{x_5}$) is obtained by taking the IGFT of individual signals defined in the spectral domain described in section II.

Since the graph smoothness constraint is applied on the coefficient $\boldsymbol{Z}$, $\boldsymbol{Z_{gen}}$ is constructed by column-wise stacking of the vertex domain signals, ($\boldsymbol{x_1}$ to $\boldsymbol{x_5}$). This is used for generating the training data, $\boldsymbol{X_{train}}$ using a pre-defined dictionary, $\boldsymbol{D_{gen}}$. The target output for regression, $\boldsymbol{Y_{train}}$ is obtained by multiplying a known weight vector, $\boldsymbol{w_{gen}}$ with $\boldsymbol{Z_{gen}}$ obtained in the previous step. The test data, $\boldsymbol{x_{test}}$, $\boldsymbol{y_{test}}$ is generated in a similar way using different values of τ (4, 2, 0.5, 1, 3).

Non-linearity is introduced in the data by taking the cube of the individual elements of the multi-variate data before subjecting it to different methods for regression analysis.

Here, the training is carried out using 100 samples. A graph of 100 vertices is constructed using (6) with every vertex carrying a multi-dimensional feature vector or time series generated synthetically. The norm of the distance measure, $dist(i, j)$ is taken to define the connection between the vertices of the graph. The graph Laplacian is computed for this graph and is normalized using (7) before being fed to the GDLR algorithm. The dictionary of size $K = 5$ atoms is learnt for both GDLR and LR. The testing is carried out using 20

ISBN 978-90-827970-1-5 © EURASIP 2018

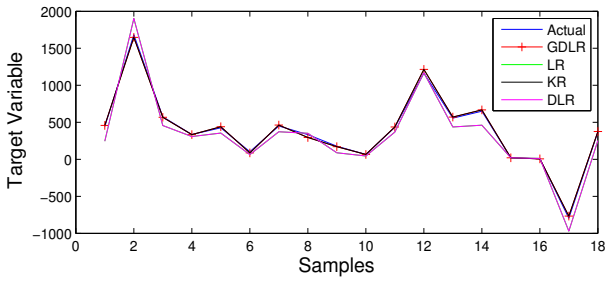

Fig. 1. Target Variable estimation with Different Methods

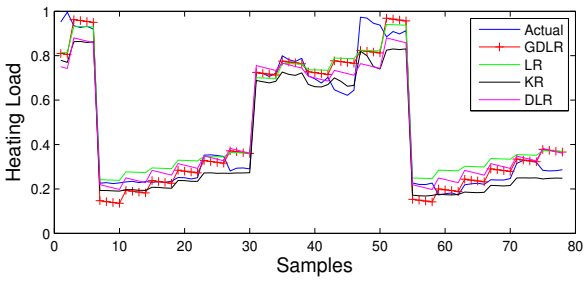

Fig. 2. Heating Load estimation with Different Methods

TABLE I
RESULTS WITH SYNTHETIC DATASET REVISED

Algorithm	MSE	MAE
GDLR($K = 5, \sigma = 20$)	**520.16**	**16.87**
KR	560	17.5
LR	674.38	19.15
DLR($K = 5$)	678.41	19.20

TABLE II
RESULTS WITH ENERGY DATASET REVISED

Algorithm	MSE	MAE
GDLR($K = 7, \sigma = 0.6$)	**0.0046**	**0.051**
KR	0.0048	0.0529
LR	0.0057	0.0607
DLR($K = 5$)	0.0058	0.054

samples of the test data. Figure 1 presents the estimation results of the target or dependent variable using different methods. Table I summarizes the estimation accuracy obtained with the test data in terms of MSE and MAE. It is observed that exploiting the underlying graph structure, GDLR results in better performance than the traditional LR and DLR methods.

B. Real-life Data

Energy Efficiency is one of the public UCI dataset which is considered for regression analysis. This dataset is comprised of 768 samples of 8 features (Relative compactness, Surface area, Wall area, Roof area, Overall height, Orientation, Glazing area, Glazing area distribution) which are used to assess the heating load and cooling load requirements of buildings. It has data from 12 different building shapes [28]. Here, the heating load is modeled as a function of the building parameters. The features are normalized before subjecting them to different regression methods. Training is carried out using 690 samples and the graph of 690 vertices is constructed in the similar way as in the case of synthetic data. The dictionary of size $K = 7$ atoms is considered for GDLR and $K = 5$ atoms for DLR which gave the best performance. It is worth noting at this juncture that the learnt dictionary coefficients $\boldsymbol{Z}$ exhibits the same smoothness over the underlying data graph characterized by $\boldsymbol{L}$. Testing is carried out on the remaining 78 samples and the estimation results using different methods are presented in Fig 2. Table 2 gives the MSE and MAE of the estimation results obtained using different methods. As is evident from the plot, GDLR is able to capture the inter-relation between different features and so it outperforms KR slightly and LR and DLR by a fair margin.

V. CONCLUSION

The paper presents an approach to build a regressor by jointly learning the graph-signal dictionary (a representation of the data) and the relevant coefficients. The initial experimentation carried out with both synthetic and the real data demonstrated the applicability of the proposed method. Further examination is underway with different graph-structured data sets towards bringing out its full potential.

ACKNOWLEDGMENT

The authors would like to thank Dr. Angshul Majumdar of IIIT Delhi for his continuous support and insightful inputs.

REFERENCES

[1] 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.
[2] I. Tosic and P. Frossard, "Dictionary learning," *IEEE Signal Processing Magazine*, vol. 28, no. 2, pp. 27–38, March 2011.
[3] C. Bao, H. Ji, Y. Quan, Z. Shen, undefined, undefined, undefined, and undefined, "Dictionary learning for sparse coding: Algorithms and convergence analysis," *IEEE Transactions on Pattern Analysis & Machine Intelligence*, vol. 38, no. 7, pp. 1356–1369, 2016.
[4] J. Mairal, J. Ponce, G. Sapiro, A. Zisserman, and F. R. Bach, "Supervised dictionary learning," in *Advances in Neural Information Processing Systems 21*, D. Koller, D. Schuurmans, Y. Bengio, and L. Bottou, Eds. Curran Associates, Inc., 2009, pp. 1033–1040.
[5] G. Chen and D. Needell, "Compressed sensing and dictionary learning," *Finite Frame Theory: A Complete Introduction to Overcompleteness*, vol. 73, 2016.
[6] D. D. Lee and H. S. Seung, "Learning the parts of objects by nonnegative matrix factorization," *Nature*, vol. 401, pp. 788–791, 1999.
[7] B. A. Olshausen and D. J. Field, "Sparse coding with an overcomplete basis set: A strategy employed by v1?" *Vision Research*, vol. 37, no. 23, pp. 3311 – 3325, 1997. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0042698997001697
[8] K. Engan, S. O. Aase, and J. H. Husoy, "Method of optimal directions for frame design," in *1999 IEEE International Conference on Acoustics, Speech, and Signal Processing. Proceedings. ICASSP99 (Cat. No.99CH36258)*, vol. 5, 1999, pp. 2443–2446 vol.5.
[9] M. Aharon, M. Elad, and A. M. Bruckstein, "K-svd and its non-negative variant for dictionary design," in *Proceedings of the SPIE conference wavelets*, 2005, pp. 327–339.
[10] H. V. Nguyen, V. M. Patel, N. M. Nasrabadi, and R. Chellappa, "Design of non-linear kernel dictionaries for object recognition," *IEEE Transactions on Image Processing*, vol. 22, no. 12, pp. 5123–5135, Dec 2013.

[11] A. Golts and M. Elad, "Linearized kernel dictionary learning," *IEEE Journal of Selected Topics in Signal Processing*, vol. 10, no. 4, pp. 726–739, June 2016.
[12] C. Peng, H. Cheng, and M. Ko, "An efficient two-stage sparse representation method," *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 30, no. 01, p. 1651001, 2016.
[13] 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.
[14] I. Tosic and P. Frossard, "Dictionary learning: What is the right representation for my signal?" *IEEE Signal Processing Magazine*, vol. 28, no. 2, pp. 27–38, 2011.
[15] R. Ganti and R. M. Willett, "Sparse Linear Regression With Missing Data," *ArXiv e-prints*, Mar. 2015.
[16] K. Kumar, A. Majumdar, G. Chandra, and A. A. Kumar, "Regressing kernel dictionary learning," in *Accepted in 2018 IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, 2018.
[17] D. Thanou, D. I. Shuman, and P. Frossard, "Learning parametric dictionaries for signals on graphs," *IEEE Transactions on Signal Processing*, vol. 62, no. 15, pp. 3849–3862, 2014.
[18] X. Zhang, X. Dong, and P. Frossard, "Learning of structured graph dictionaries," in *Acoustics, Speech and Signal Processing (ICASSP), 2012 IEEE International Conference on*. IEEE, 2012, pp. 3373–3376.
[19] Y. Yankelevsky and M. Elad, "Graph-constrained supervised dictionary learning for multi-label classification," in *Science of Electrical Engineering (ICSEE), IEEE International Conference on the*. IEEE, 2016, pp. 1–5.
[20] ——, "Dual graph regularized dictionary learning," *IEEE Transactions on Signal and Information Processing over Networks*, vol. 2, no. 4, pp. 611–624, Dec 2016.
[21] K. N. Ramamurthy, J. J. Thiagarajan, P. Sattigeri, and A. Spanias, "Learning dictionaries with graph embedding constraints," in *2012 Conference Record of the Forty Sixth Asilomar Conference on Signals, Systems and Computers (ASILOMAR)*, Nov 2012, pp. 1974–1978.
[22] X. Dong, D. Thanou, P. Frossard, and P. Vandergheynst, "Laplacian matrix learning for smooth graph signal representation," in *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, April 2015, pp. 3736–3740.
[23] H. Zheng, G. Cheung, and L. Fang, "Analysis of sports statistics via graph-signal smoothness prior," in *2015 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA)*, Dec 2015, pp. 1071–1076.
[24] D. I Shuman, S. K. Narang, P. Frossard, A. Ortega, and P. Vandergheynst, "The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains," *IEEE Signal Processing Magazine*, vol. 30, no. 3, pp. 83–98, May 2013.
[25] R. Bhatia, *Matrix Analysis*. Springer, 1997, vol. 169.
[26] Y. Cao, "Multivariant Kernel Regression and Smoothing," https://in.mathworks.com/matlabcentral/fileexchange/19279, March 2008.
[27] N. Perraudin, J. Paratte, D. I. Shuman, V. Kalofolias, P. Vandergheynst, and D. K. Hammond, "GSPBOX: A toolbox for signal processing on graphs," *CoRR*, vol. abs/1408.5781, 2014. [Online]. Available: http://arxiv.org/abs/1408.5781
[28] A. X. A. Tsanas, "Accurate quantitative estimation of energy performance of residential buildings using statistical machine learning tools," in *Energy and Buildings*, vol. 49, 2012, pp. 560–567.

ISBN 978-90-827970-1-5