SPARSE REPRESENTATION AND PIECE-WISE LINEAR KERNEL

Shiro Ikeda

The Institute of Statistical Mathematics
4-6-7 Minami-Azabu, Minatoku, Tokyo 106-8569, Japan

ABSTRACT

We propose a new type of kernel function, where feature space is explicitly given with a piece-wise linear mapping from the input space. This idea is inspired by sparse linear system analysis, where inputs are represented as a sparse linear combination of "dictionary vectors." This article gives the idea of such kernel function, and some preliminary experimental results.

1. INTRODUCTION

Although kernel methods became one of the standard methods especially in machine learning, there are still a lot of open problems. One important problem is the choice of kernel function. In many applications, we replace this problem by tuning some parameters which characterize the kernel functions, for example, width of Gaussian kernel or order of polynomial kernel, but it is important to find other families of kernel functions. In this article, we propose a new type of kernel function. The idea comes from the sparse or over-complete (under-determine) linear system analysis.

Recently, the idea of sparse linear system is attracting a lot of interests. Olshausen and Field[1] have clearly shown that the sparse representation principle describes the information representation in visual cortex. For speech signals, recovering multiple source signals from smaller number of sensors' observations is important [2, 3], where the sparsity of sound signals in time or time-frequency domain plays an important role. Independently, Donoho and Huo discussed conditions for sparse representation to be unique [5], and the lasso (least absolute shrinkage and selection operator) model proposed by Tibishirani[6] is also related to the sparse representation.

In sparse linear system, data are represented as a linear combination of over-complete basis vectors, which we call dictionary. There are a lot of works which study the algorithm of dictionary learning and the performance of source extraction[7, 8, 9, 10, 11].

In this article, we view the sparse linear system from different point. We consider the sparse representation as the features of the input signals. Since the features are sparse, we have a good analogy with kernel methods, where we bring inputs to high-, even infinite-dimensional feature space. We treat the sparse representation as the feature vector and propose a kernel function. This method can be viewed as a piece-wise linear operation, therefore we call this piece-wise linear kernel. We first give the outline of the sparse linear system analysis, and propose a piece-wise linear kernel. We also give a preliminary experimental results.

2. SPARSE LINEAR SYSTEM ANALYSIS

2.1. Problem

Consider the case where data are nonzero N-dimensional real vector $\boldsymbol{x} = (x_1, \cdots, x_N) \in \Re^N$, $\boldsymbol{x} \neq \mathbf{o}$. Our problem is to find a representation of $\boldsymbol{x}$ as a linear combination of a set of basis vectors, which we call dictionary. Let $D \in \Re^{N\times M}$ defined as,

$$D = [\boldsymbol{d}_1, \cdots, \boldsymbol{d}_M], \quad \boldsymbol{d}_k \in \Re^N.$$

In this paper, we consider the case where $M > N$. Now, our problem is to find the following sparse vector $\boldsymbol{s}$,

$$\boldsymbol{x} = \boldsymbol{d}_1 s_1 + \cdots + \boldsymbol{d}_M s_M = D\boldsymbol{s}, \quad \boldsymbol{s} = (s_1, \cdots, s_M)^T. \tag{1}$$

There is another model which assume additive noise as

$$\boldsymbol{x} = \boldsymbol{d}_1 s_1 + \cdots + \boldsymbol{d}_M s_M + \boldsymbol{n}, \quad \boldsymbol{n} = \Re^N, \boldsymbol{n} \sim p(\boldsymbol{n}). \tag{2}$$

where noise distribution is given as $p(\boldsymbol{n})$. One widely used distribution is multi-dimensional Gaussian distribution [1, 8, 9]. Both are used to derive the sparse representation where model in eq.(2) may give sparser representation. In this article, we use noiseless model in eq.(1). We further impose the following two conditions, on dictionary, that is

$$\begin{aligned} \|\boldsymbol{d}_k\|_2 = 1, &\quad k = 1, \cdots, M \\ \mathrm{rank}[\boldsymbol{d}_{i_1}, \cdots, \boldsymbol{d}_{i_N}] = N, &\quad i_l \neq i_k, \\ &\quad i_1, \cdots, i_N \in \{1, \cdots, M\}, \end{aligned} \tag{3}$$

where $\|\cdot\|_2$ denote l_2 norm. We restrict every dictionary vector to stay on the surface of unit sphere in $\Re^N$, and any N combination of the vectors are linearly independent.

This work was supported by the Grant-in-Aid for Scientific Research 16700227, MEXT, Japan.

Equation(1) has infinite solutions, and our goal is to find the "sparse" solution among them. Following the definitions of Donoho and Elad[4], we have two natural formulations,

$$\begin{array}{llll} (P_0) & \text{Minimize} & \|\boldsymbol{s}\|_0, & \text{subject to} \quad \boldsymbol{x} = D\boldsymbol{s} \\ (P_1) & \text{Minimize} & \|\boldsymbol{s}\|_1, & \text{subject to} \quad \boldsymbol{x} = D\boldsymbol{s} \end{array}$$

Here, the l_0 norm $\|\cdot\|_0$ gives the number of nonzero elements of the vector, while l_1 norm $\|\cdot\|_1$ gives the sum of absolute values of the elements.

It seems the solution of (P_0) gives the "sparse" solution. Donoho and Elad studied the condition where the solution of (P_0) is unique and identical to that of (P_1) [4], but it can only happen when $\|s\|_0 < N$, because if $\|s\|_0 = N$, there are multiple solutions to (P_0). Moreover there is no efficient algorithm to solve (P_0), and we focus on (P_1) where the problem is solved efficiently with LP (linear programming).

2.2. Algorithm

We briefly explain the LP formulation to solve (P_1). Let us define the following two positive vectors,

$$\boldsymbol{u} = (u_1, \cdots, u_N)^T, \quad \boldsymbol{v} = (v_1, \cdots, v_N)^T, \quad \boldsymbol{u}, \boldsymbol{v} \in \Re^N_+.$$

and let $\boldsymbol{s} = \boldsymbol{u} - \boldsymbol{v}$. Now, (P_1) is rewritten as

$$(P_1') \qquad \text{Minimize} \quad \sum_i (u_i + v_i),$$

$$\text{subject to } \boldsymbol{x} = \tilde{D}\begin{pmatrix}\boldsymbol{u}\\ \boldsymbol{v}\end{pmatrix}, \ \boldsymbol{u}, \boldsymbol{v} \succeq 0. \text{ where } \tilde{D} = [D, -D].$$

Here, $\boldsymbol{u}, \boldsymbol{v} \succeq 0$ denotes that every component of $\boldsymbol{u}$ and $\boldsymbol{v}$ is non-negative. This is a standard LP formulation, and under the condition in eq.(3), the sparse solution of (P_1) is unique[10]. Let us define $\boldsymbol{y} = (\boldsymbol{u}^T, \boldsymbol{v}^T)^T \in \Re^{2M}_+$, and the solution of (P_1') as $\hat{\boldsymbol{y}}$. The solution $\hat{\boldsymbol{y}}$ can be efficiently computed with simplex method. It is known that at most N components of $\hat{\boldsymbol{y}}$ are not zero, that is $\|\hat{\boldsymbol{y}}\|_0 \leq N$.

2.3. Geometrical view

From our assumption in eq.(3), any N columns of D are linearly independent. Let us choose N columns from $\tilde{D}$, and let the matrix A. We also define B as a $N \times (2M - N)$ matrix which consists of the rest of the columns. By rearranging $\boldsymbol{y}$ as $\boldsymbol{y} \rightarrow (\boldsymbol{y}_A^T, \boldsymbol{y}_B^T)^T$, corresponding to A and B, and we have the following relation

$$\boldsymbol{x} = \tilde{D}\boldsymbol{y} = A\boldsymbol{y}_A + B\boldsymbol{y}_B.$$

Since A is invertible,

$$\boldsymbol{y}_A = A^{-1}\boldsymbol{x} - A^{-1}B\boldsymbol{y}_B.$$

It is known [11] that optimal solution is expressed as $\boldsymbol{y}_A = A^{-1}\boldsymbol{x}, \boldsymbol{y}_B = \mathbf{o}$, for the A, B which satisfies

$$A^{-1}\boldsymbol{x} \succeq 0, \quad \mathbf{1}^T_{(2M-N)} - \mathbf{1}^T_N A^{-1}B \succeq \mathbf{o}^T.$$

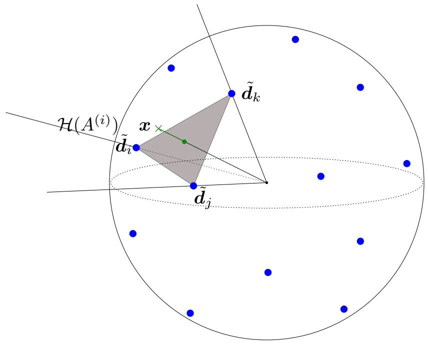
$\mathcal{H}(A^{(i)})$
$\boldsymbol{x}$
$\tilde{\boldsymbol{d}}_i$
$\tilde{\boldsymbol{d}}_j$
$\tilde{\boldsymbol{d}}_k$

Fig. 1. Geometrical view of (P_1') solution ($N = 3$): blue circles correspond to dictionary vectors on the surface of sphere, while green cross corresponds to data point $\boldsymbol{x}$.

Following the discussion in [11], we further define a set of possible optimal matrices A as

$$\begin{aligned} \mathcal{A} &= \{A | \mathbf{1}^T_{(2M-N)} - \mathbf{1}^T_N A^{-1}B \succeq \mathbf{o}^T, \det(A) \neq 0\} \\ &= \{A^{(1)}, \cdots, A^{(q)}\}. \end{aligned}$$

$\mathcal{A} = \{A^{(i)}\}$ splits the whole space of $\{\boldsymbol{x}\}$ into q disjoint convex hulls defined as

$$\mathcal{H}(A^{(i)}) = \{\boldsymbol{x} | A^{-1}\boldsymbol{x} \succeq 0\},$$

Figure 1 schematically shows the problem for $N = 3$. The solution of (P_1') is viewed as follows: the whole space is split into q disjoint region $\mathcal{H}(A^{(1)}), \cdots, \mathcal{H}(A^{(q)})$ and in each region, $\boldsymbol{x}$ is mapped to a subspace of a higher dimensional space by a linear transform given by ${A^{(i)}}^{-1}$. Therefore, this transformation from $\boldsymbol{x}$ to $\boldsymbol{y}$ is a piece-wise linear transformation.

Furthermore, the transformation is continuous. We briefly sketch the proof. As far as $\boldsymbol{x}$ stays inside a $\mathcal{H}(A^{(i)})$, mapping from $\boldsymbol{x}$ to $\boldsymbol{y}$ is linear and continuous. When we move $\boldsymbol{x}$ from one $\mathcal{H}(A^{(i)})$ to adjacent $\mathcal{H}(A^{(j)})$, $\boldsymbol{x}$ goes through a boundary. At the boundary of $\mathcal{H}(A^{(i)})$ and $\mathcal{H}(A^{(j)})$, they share some dictionary vectors. Let $\tilde{\boldsymbol{x}} \in \mathcal{H}(A^{(i)}) \cap \mathcal{H}(A^{(j)})$, $\tilde{\boldsymbol{x}} \neq \mathbf{o}$ and consider we move $\boldsymbol{x}$ from $\mathcal{H}(A^{(i)})$ to $\mathcal{H}(A^{(j)})$ through $\tilde{\boldsymbol{x}}$. When $\boldsymbol{x} \in \mathcal{H}(A^{(i)})$, the transformation from $\boldsymbol{x}$ to $\boldsymbol{y}$ is defined as ${A^{(i)}}^{-1}\boldsymbol{x}$, and when it becomes $\tilde{\boldsymbol{x}}$, ${A^{(i)}}^{-1}\tilde{\boldsymbol{x}}$ only have positive values on the coefficients, which corresponds to the dictionary vectors included in $\mathcal{H}(A^{(i)}) \cap \mathcal{H}(A^{(j)})$. This is the same when $\boldsymbol{x}$ moves from $\mathcal{H}(A^{(j)})$ to $\tilde{\boldsymbol{x}}$, and the transformation from $\boldsymbol{x}$ to $\boldsymbol{y}$ is piece-wise linear and continuous.

3. MAPPING AND KERNELS

3.1. Features

Now, let us define $\boldsymbol{\phi}_{\tilde{D}}(\boldsymbol{x})$ as a mapping from $\boldsymbol{x}$ to $\boldsymbol{y}$,

$$\boldsymbol{\phi}_{\tilde{D}}(\boldsymbol{x}) = \boldsymbol{y}(\boldsymbol{x}) = \underset{\boldsymbol{s}}{\operatorname{argmin}} \Big\{ \sum_{i=1}^{2M} y_i \;\Big|\; \boldsymbol{x} = \tilde{D}\boldsymbol{y}, \boldsymbol{y} \succeq 0 \Big\}. \quad (4)$$

As it is shown in the last section, $\boldsymbol{\phi}_{\tilde{D}}(\boldsymbol{x})$ is a piece-wise linear and continuous mapping. We consider this $\boldsymbol{\phi}_{\tilde{D}}(\boldsymbol{x})$ as a feature vector. We define a kernel function as,

$$K(\boldsymbol{x}, \boldsymbol{z}) = \boldsymbol{\phi}_{\tilde{D}}(\boldsymbol{x})^T \boldsymbol{\phi}_{\tilde{D}}(\boldsymbol{z}).$$

The Gram matrix $G = [K(\boldsymbol{x}_i, \boldsymbol{x}_j)]$ is positive semi-definite. It has an interesting property, that is, for $\boldsymbol{x} \in \mathcal{H}(A^{(i)})$ and $\boldsymbol{z} \in \mathcal{H}(A^{(j)})$, $\boldsymbol{x}, \boldsymbol{z} \neq \mathbf{o}$
if $\left(\mathcal{H}(A^{(i)}) \cap \mathcal{H}(A^{(j)})\right) \setminus \{\mathbf{o}\} = \emptyset$,

$$K(\boldsymbol{x}, \boldsymbol{z}) = 0$$

if $\left(\mathcal{H}(A^{(i)}) \cap \mathcal{H}(A^{(j)})\right) \setminus \{\mathbf{o}\} \neq \emptyset$

$$K(\boldsymbol{x}, \boldsymbol{z}) = \boldsymbol{\phi}_{\tilde{D}}(\boldsymbol{x})^T \boldsymbol{\phi}_{\tilde{D}}(\boldsymbol{z}) > 0,$$

if $\boldsymbol{x}, \boldsymbol{z} \in \mathcal{H}(A^{(i)})$,

$$K(\boldsymbol{x}, \boldsymbol{z}) = \boldsymbol{x}^T ({A^{(i)}}^{-1})^T {A^{(i)}}^{-1} \boldsymbol{z}.$$

These properties are distinctive from Gaussian kernel function, where $K(\boldsymbol{x}, \boldsymbol{z}) > 0$ for $\forall \boldsymbol{x}, \boldsymbol{z} \in \Re^N$. Note that Gaussian kernel provides a smooth function, while piece-wise linear kernel provides a continuous but angular function.

If this kernel is used for SVM, we will have a linear function of $f(\boldsymbol{x}) = \boldsymbol{w}^T \boldsymbol{\phi}_{\tilde{D}}(\boldsymbol{x}) + b$ whose sign of the output gives the estimated class. Since the mapping is linear in each convex hull $\mathcal{H}(A^{(i)})$, we only have single linear separator in each convex hull $\mathcal{H}(A^{(i)})$. This seems very restrictive. But there are a lot of cases where this kernel is effective. For example, in image recognition, or text classification, the data can be normalized, and only the ratio, or the direction of the data is important. In such a case. This kernel is might be effective.

3.2. Dictionary

We have defined the piece-wise linear kernel. When the dictionary vectors are defined, the idea is natural. Now, it is not difficult to imagine that the performance of the resulting method based on the piece-wise linear kernel is strongly affected by the choice of dictionary vectors.

Let us consider the problem of learning dictionary vectors from training data. There are a lot of works on dictionary learning [7, 8, 12, 9, 10, 1]. One popular method is to define a noisy model as in eq.(2) and the prior of $\boldsymbol{s}$ is given, for example, as a bilateral exponential distribution. Then using the MAP estimation of $\boldsymbol{y}$, the likelihood is defined, and it is used as the cost function to learn the dictionary vectors. This is an interesting direction, and we may use them in our future experiments. However, in this article, we show our preliminary results where the data size is quite small, and we used simpler methods for dictionary learning. We briefly explain them.

Random dictionary Let us assume that the data are centerized. Then, one naive method of creating the dictionary vectors is to generate random N dimensional vectors on the surface of unit sphere.

Sample vector dictionary It is also possible to use the normalized training data points or a subset of them as the dictionary vectors. Given number M of the vectors are chosen randomly from the training data set.

k–mean Li, et al., suggested to use k–mean method [10] for dictionary learning. First, the training data are centerized and normalized to have unit length. For the normalized data, k-mean method is applied where the number of the dictionary vectors k is prefixed, and finally those k–mean vectors are normalized to have a unit length.

We used above three dictionary learning methods in the next section. Note that in order to learn the dictionary vectors, the number of dictionary vectors M, must be specified beforehand. Unfortunately, we have not used any learning method to determine M, but tried many numbers.

4. EXPERIMENT

We applied this piece-wise linear kernel for classification problem, with SVM. We used Pima Indians diabetes database (8 numeric attributes, no missing values, 2 classes, 768 instances) and Ionosphere database (34 numeric attributes, no missing values, 2 classes, 351 instances) from UCI Machine Learning Repository.

For both data sets, first 200 instances are used for training, and the rests for testing. Mean of each attribute is computed from training data, and extracted from all the data. Also variance of each attribute is computed from training data, and every attribute of data is divided with corresponding standard deviation.

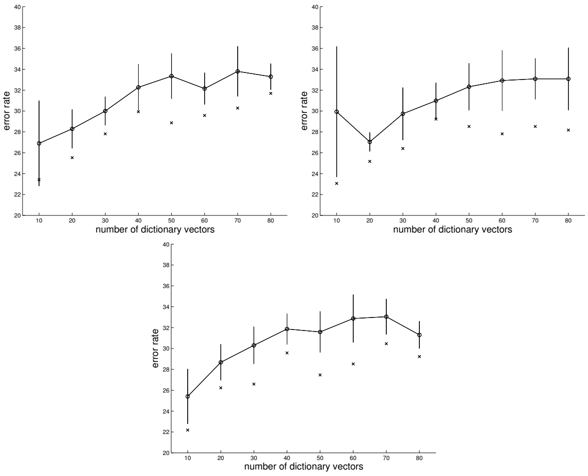

Fig. 2. Error rate of each experiments on Pima Indian diabetes database: left top, random dictionary is used, right top, sample vector dictionary is used, and bottom, k–mean dictionary is used.

First, we show the results for Pima Indians diabetes database. We tried three methods in the last session for dictionary learning. The number of the dictionary vectors varies from 10 to 80, and for each method and each number of vectors, we repeated same experiments for 10 times, since each dictionary learning methods includes random process. Figure 2 shows the error rates of the experiments. In the figure, we show the mean of the errors with standard deviation. Also the minimum error rate attained in each 10 experiments was plotted with crosses. The minimum error rate 22.2% was attained when we used k–means method for dictionary learning, and the number of the dictionary vectors is 10.

For comparison Gaussian kernel function was used, where its width parameter was tuned to 30, and the error rate was less than 21.4% .

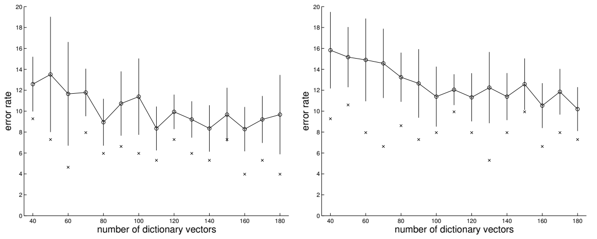

Fig. 3. Error rate of each experiments on Ionosphere database: left, random dictionary is used, right, sample vector dictionary is used.

Next, we show the results for Ionosphere database. We have applied random dictionary and sample dictionary methods, since k–mean learning needed much time in this case. The number of the vectors varied from 40 to 180.

The result is shown in Fig.3. The minimum error rate was attained when we used the random dictionary with size of 180, and its error rate was 4.0%. For comparison, we note that Gaussian kernel with 5 as its width parameter attain error rate of 2.0%.

5. DISCUSSION AND CONCLUSION

We have shown that the idea of the sparse linear system can be connected to kernel method. This new kernel function works as a piece-wise linear function. The two experiments show that the performance strongly depends on the selection of dictionary vectors, but if we choose a good set of dictionary vectors, it may as good as commonly used kernels.

Although we have to work more in order to obtain convincing results, this kernel function has a different characteristics from well-know kernels, and it is important to show its potential. Moreover, sparse representation with overcomplete basis is a well studied subject, and it is important to show it can be naturally connected to kernel method.

6. REFERENCES

[1] B. A. Olshausen and D. J. Field, “Emergence of simple-cell receptive field properties by learning a sparse code for natural images,” *Nature*, vol. 381, pp. 607–609, 1996.

[2] P. Bofill and M. Zibulevski, “Underdetermined blind source separation using sparse representation,” *Signal Processing*, vol. 81, pp. 2353–2362, 2001.

[3] T-W. Lee, M. S. Lewicki, M. Girolami, and T. J. Sejnowski, “Blind source separation of more sources than mixtures using overcomplete representations,” *IEEE Signal Processing Letters*, vol. 6, no. 4, pp. 87–90, Apr. 1999.

[4] D. L. Donoho and M. Elad, “Optimally sparse representation in general (nonorthogonal) dictionaries via l^1 minimization,” *Proc. Nat. Aca. Sci.*, vol. 100, no. 5, pp. 2197–2202, Mar. 2003.

[5] D. L. Donoho and X. Huo, “Uncertainty principles and ideal atomic decomposition,” *IEEE Trans. Inform. Theory*, vol. 47, no. 7, pp. 2845–2862, Nov. 2001.

[6] R. Tibishirani, “Regression shrinkage and selection via the lasso,” *J. R. Stat. Soc., Series B*, vol. 58, no. 1, pp. 267–288, 1996.

[7] A. Hyvärinen, “Estimation of non-normalized statistical models by score matching,” *Journal of Machine Learning Research*, vol. 6, pp. 695–709, 2005.

[8] A. Hyvärinen and M. Inki, “Estimating overcomplete independent component bases for image windows,” *Journal of Math. Imaging and Vision*, vol. 17, pp. 139–152, 2002.

[9] M. S. Lewicki and T. J. Sejnowski, “Learning overcomplete representations,” *Neural Comput.*, vol. 12, no. 2, pp. 337–365, Feb. 2000.

[10] Y. Li, A. Cichocki, and S. Amari, “Analysis of sparse representation and blind source separation,” *Neural Comput.*, vol. 16, no. 6, pp. 1193–1234, Jun. 2004.

[11] I. Takigawa, M. Kudo, and J. Toyama, “Performance analysis of minimum l_1-norm solutions for underdetermined source separation,” *IEEE Trans. Signal Processing*, vol. 52, no. 3, pp. 582–591, Mar. 2004.

[12] K. Kreutz-Delgado, J. F. Murray, B. D. Rao, K. Engan, T-W. Lee, and T. J. Sejnowski, “Dictionary learning algorithm for sparse representation,” *Neural Comput.*, vol. 15, no. 2, pp. 349–396, 2003.