Time signal filtering by relative neighborhood graph localized linear approximation

Sørensen, John Aasted

Published in:
Proceedings of the 4th IEEE Workshop Neural Networks for Signal Processing

Link to article, DOI:
10.1109/NNSP.1994.366051

Publication date:
1994

Document Version
Publisher's PDF, also known as Version of record

Time Signal Filtering by Relative Neighborhood Graph Localized Linear Approximation.

John Aasted Sørensen, Electronics Institute, Build. 349
Technical University of Denmark, 2800 Lyngby Denmark.

Abstract. A time signal filtering algorithm based on the relative neighborhood graph (RNG) used for localization of linear filters is proposed. The filter is constructed from a training signal during two stages. During the first stage an RNG is constructed. During the second stage, localized linear filters are associated each RNG node and adapted to the training signal. The filtering of a test signal is then carried out by inserting the test signal vectors in the RNG followed by the determination of the filter output as a function of the linear filters of the RNG nodes to which the vectors are associated. Training examples are given on a segment of a speech signal and a signal with burst structure generated from a bilinear Subba Rao model.

1 Introduction

A time signal filtering algorithm based on relative neighborhood graph (RNG) localized linear filters is proposed. The filter is constructed during two stages:
During the first stage, a training signal x_n, $n = 1, \ldots, N$ is used for generation of an RNG using an input dimension D. The RNG of a set of vectors, connect the vectors $\mathbf{x}_i^T = (x_i, \ldots, x_{i-D+1})$ and $\mathbf{x}_j$ if the intersection of the spheres with radii equal to the distance between $\mathbf{x}_i$ and $\mathbf{x}_j$ and centered in $\mathbf{x}_i$ and $\mathbf{x}_j$ does not contain any vector from the set. This intersection is also denoted the lune $\Lambda_{i,j}$ of $\mathbf{x}_i$ and $\mathbf{x}_j$. A lune thus represents a part of the input space which is mainly defined by the two vectors generating the lune. The result of the first stage is a structural representation of the input space based on the RNG. This structure is then used for localizing linear filters, adapted by a gradient algorithm to the training set during the second stage.

The filtering of a test signal t_n, $n = 1, \ldots$ is then carried out as follows: Insert test vectors $\mathbf{t}_n^T = (t_n, t_{n-1}, \ldots, t_{n-D+1})$ into the RNG, by determining all the lunes to which each $\mathbf{t}_n$ belongs. These lunes defines the neighborhood of $\mathbf{t}_n$. The filter output is then a function of the linear filters belonging to this neighborhood. In the example hereafter the filter output function is a weighted mean value of the neighborhood filters.

0-7803-2026-3/94 $4.00 © 1994 IEEE

The Relative Neighborhood Graph (RNG)

If the open sphere with center in $\mathbf{x}$ and radius r is denoted

$$B(\mathbf{x}, r) = \{\mathbf{y} \mid d(\mathbf{x}, \mathbf{y}) < r\} \tag{1}$$

where $d(\mathbf{x}, \mathbf{y})$ is the distance between $\mathbf{x}$ and $\mathbf{y}$,
then the lune $\Lambda_{i,j}$ of $\mathbf{x}_i$ and $\mathbf{x}_j$ is determined by

$$\Lambda_{i,j} = B(\mathbf{x}_i, d(\mathbf{x}_i, \mathbf{x}_j)) \cap B(\mathbf{x}_j, d(\mathbf{x}_i, \mathbf{x}_j)) \tag{2}$$

or by

$$\Lambda_{i,j} = \{\mathbf{x} \mid \max(d(\mathbf{x}_i, \mathbf{x}), d(\mathbf{x}, \mathbf{x}_j)) < d(\mathbf{x}_i, \mathbf{x}_j)\} \tag{3}$$

The lune is exemplified in Figure 1.

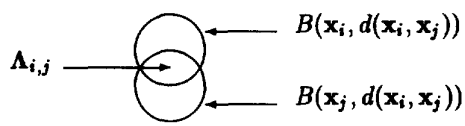

Figure 1: The relative neighborhood $\Lambda_{i,j}$.

Based on this definition of a lune [1], the RNG of an input signal x_n, $n = 1, \ldots, N$ is determined by

$$[\mathbf{P}, \mathbf{C}] = RNG(x_n,\ n = 1, \ldots, N,\ D) \tag{4}$$

where

D : Dimension of input space.
$\mathbf{P}$ is a matrix of RNG nodevectors.
$\mathbf{P} = [\mathbf{p}_1, \mathbf{p}_2, \ldots, \mathbf{p}_R] \in \mathbf{R}^{D \times R}$
R is the number of nodes in the RNG.
$\mathbf{C}$ is the incidence matrix of the RNG.
$\mathbf{C} = [\mathbf{c}_1, \mathbf{c}_2, \ldots, \mathbf{c}_R] \in \{0, 1\}^{R \times R}$
$\mathbf{c}_i^T = (c_{1,i}, \ldots, c_{R,i})$
$c_{i,j} = 1$ if $\Lambda_{\mathbf{p}_i, \mathbf{p}_j}$ is empty, otherwise $c_{i,j} = 0$.

2 Training Algorithm

The training algorithm is divided into 2 stages.

Stage 1: Generation of the RNG filter structure.
In the first stage the RNG is determined according to

$$[\mathbf{P}, \mathbf{C}] = RNG(\boldsymbol{x}_n,\ n = 1, \ldots, N) \tag{5}$$

Stage 2: Adaptation of RNG localized linear filters.

The RNG localized linear filters are now formed by associating a FIR filter with each node of the RNG. This leads to the following filter matrix

$$\mathbf{W} = [\mathbf{w}_1, \ldots, \mathbf{w}_R] \tag{6}$$

where $\mathbf{w}_k^T = [w_{1,k}, \ldots, w_{D,k}, w_{D+1,k}]$. The term $w_{D+1,k}$ is the bias of the RNG node filter number $k = 1, \ldots, R$.

Assuming that the current augmented input signal vector at time step n is $\mathbf{z}_n^T = (x_n, x_{n-1}, \ldots, x_{n-D+1}, 1)$ and the augmented RNG node vectors are $\mathbf{r}_j^T = [\mathbf{p}_j^T, 0]$ for $j = 1, \ldots, R$, gives the following filter output, using the weighted mean of neighborhood

$$\hat{x}_n = \sum_{j=1}^{R} \frac{\gamma_{j,n}}{\gamma_n} \mathbf{w}_j^T (\mathbf{z}_n - \mathbf{r}_j) \tag{7}$$

Here $\gamma_{j,n}$ is the number of times the RNG node number j is a member of a lune to which $\mathbf{x}_n$ belongs, when $\mathbf{x}_n$ is inserted into the RNG. $\mathbf{x}_n$ is inserted in the RNG by determining the lunes to which $\mathbf{x}_n$ belongs. The total number of nodes in the lunes to which $\mathbf{x}_n$ belongs is

$$\gamma_n = \sum_{j=1}^{R} \gamma_{j,n} \tag{8}$$

The RNG node weighting matrix at time step n becomes

$$\mathbf{\Gamma}_n = \begin{bmatrix} \frac{\gamma_{1,n}}{\gamma_n} & & 0 \\ & \ddots & \\ 0 & & \frac{\gamma_{R,n}}{\gamma_n} \end{bmatrix} \tag{9}$$

In (7) $\hat{x}$ is formed as a weighted mean value of the predictions from the nodes which constitute the lunes to which $\mathbf{x}_n$ belongs. Defining the error vector between the current input vector $\mathbf{x}_n$ and RNG node number j gives $\delta_j = \mathbf{z}_n - \mathbf{r}_j$ for $j = 1, \ldots, R$. This defines the input signal matrix at time step n to the RNG nodes:

$$\mathbf{\Delta}_n = [\delta_1, \ldots, \delta_R] \tag{10}$$

From (7), (9) and (10) the filter output can be represented

$$\hat{x}_n = trace(\mathbf{W}_n^T \mathbf{\Delta}_n \mathbf{\Gamma}_n) \tag{11}$$

Using the LMS adaptation of the filter matrix $\mathbf{W}_n$, where the index n denotes the time step, leads to:

$$\mathbf{W}_{n+1} = \mathbf{W}_n + \mu e_n \mathbf{\Delta}_n \mathbf{\Gamma}_n \tag{12}$$

where $e_n = x_n - \hat{x}_n$ is the prediction error and μ is the adaptation constant.

3 Training Experiments

The training algorithm is exemplified on a speech signal segment shown in Figure 2 and on a segment of the bilinear model of Subba Rao [2]:

$$x_n = 0.8x_{n-1} - 0.4x_{n-2} + 0.6x_{n-1}e_{n-1} + 0.7x_{n-2}e_{n-1} + e_n \tag{13}$$

where e_n is white, Gaussian noise with variance 1. As shown in Figure 3, this signal exhibits burst structure.

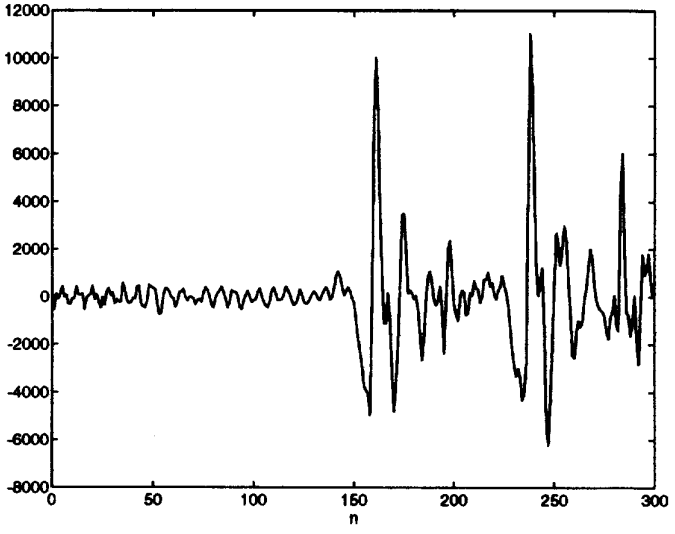

Figure 2: Speech input signal.

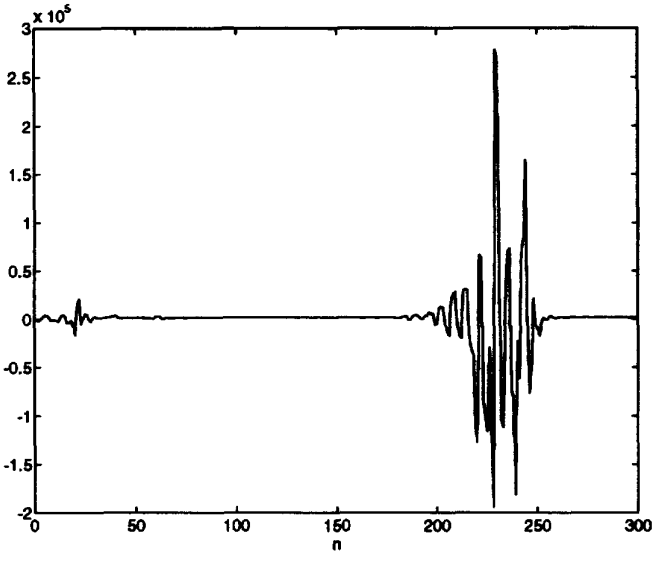

Figure 3: Input signal from the bilinear Subba Rao model.

The predictions are evaluated by the normalized, mean square error [3]

$$NMSE(x_n, \hat{x}_n) = \frac{1}{\sigma^2 N} \sum_{n=1}^{N} (x_n - \hat{x}_n)^2 \tag{14}$$

where x_n is the true value of the input signal at time step n and $\hat{x}_n$ is the predicted value. σ^2 is the variance of the input signal with N samples. Thus $NMSE$ is the ratio of the mean squared errors of the filtering method being trained and a method which predicts the mean at every time step.

In Table 1 are given training examples using the above speech and bilinear signal segments. The training is carried out on three models: The block linear filtering, the linear k-nearest neigbor filtering [4] and the relative neighborhood graph based filtering. From this it is seen that the training performance of the RNG filter structure is comparable to the performance of the k-nearest neighbor filtering at the same dimension of the input space. Furthermore it is expected that the RNG leads to a more suitable definition of neighborhood for localized filtering compared to the k-nearest nigborhood.

Filter	D	nn	R	NMSE speech	R	NMSE bilinear
Linear	12			0.137		0.691
k-nn	4	10		0.080		
	6	15		0.073		
	12	15		0.005		
	3	6				0.155
	4	6				0.032
	4	10				0.210
RNG	3		54	0.130	55	0.042
	4		66	0.051	58	0.0089

Table 1: Training results for the speech and the bilinear signal segment.
D: Dimension of input vector.
nn: The number of nearest neighbors in k-nearest neighbor.
R: The number of nodes in the RNG, determined in the first stage.

References

[1] Jerzy W. Jaromczyk, Godfried T. Tousssaint, *Relative Neighborhood Graphs and Their Relatives*, Proceedings of IEEE, Vol. 80, No. 9, September 1992.

[2] M.B. Priestly, *Non-linear and Non-stationaryTime series Analysis*, Academic Press, 1988.

[3] *Time Series Prediction*
Andreas S. Weigend, Neil A. Gershenfeld, Eds.
Proceedings Volume XV, Santa Fe Institute
Addison-Wesley Publishing Company, 1994.

[4] J.D. Farmer, J.J. Sidorowich
Exploiting Chaos to Predict the Future and Reduce Noise.
Tech. Rep. LA-UR-88-901, Los Alamos National laboratory, 1988.