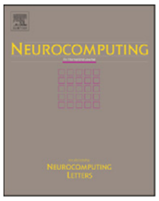
Neurocomputing 90 (2012) 59–65

Contents lists available at SciVerse ScienceDirect

Neurocomputing

journal homepage: www.elsevier.com/locate/neucom



Adaptive kernel smoothing regression for spatio-temporal environmental datasets

Federico Montesino Pouzols [a,*,1], Amaury Lendasse [b,c,d]

[a] Department of Biosciences, University of Helsinki Biocenter 3, Viikinkaari 1, PO Box 65, FI-00014 University of Helsinki, Finland
[b] Department of Information and Computer Science, Aalto University, Espoo, Finland
[c] IKERBASQUE, Basque Foundation for Science, 48011 Bilbao, Spain
[d] Computational Intelligence Group, Computer Science Faculty, University of the Basque Country, Paseo Manuel Lardizabal 1, Donostia/San Sebastián, Spain

ARTICLE INFO

Available online 23 March 2012

Keywords:
Kernel smoothing regression
Adaptive regression
Vector quantization
Spatio-temporal models
Environmental applications
Evolving intelligent systems

ABSTRACT

A method for performing kernel smoothing regression in an incremental, adaptive manner is described. A simple and fast combination of incremental vector quantization with kernel smoothing regression using adaptive bandwidth is shown to be effective for online modeling of environmental datasets. The approach proposed is to apply kernel smoothing regression in an incremental estimation of the (evolving) probability distribution of the incoming data stream rather than the whole sequence of observations. The method is illustrated on publicly available datasets corresponding to the Tropical Atmosphere Ocean array and the Helsinki Commission hydrographic database for the Baltic Sea.

1. Introduction

We describe a method for performing kernel smoothing regression in an adaptive manner. The aim of this work is to define efficient, incremental and adaptive regression methods that can be applied sequentially to data streams of incoming observations of continuous data. The motivation for this work is the need for simple and efficient regression methods that can cope with large, diverse and evolving datasets in applications in biological and environmental sciences.

The idea of adaptive regression has been explored in different contexts and a large number of methods for both linear and nonlinear regression are well established in different fields of computer science. For example, the multivariate adaptive regression splines (MARS) method [1–3] builds models as a summation of weighted basis functions following a divide and conquer strategy that aims to adapt locally. However, most research efforts so far have concentrated on offline regression.

Biological and environmental sciences have seen a great deal of development and attention over the last few decades. An impressive improvement in observational capabilities and measurement procedures has led to large databases and online monitoring systems. Biological and environmental datasets are normally defined by either regular or irregular spatial fields that can be three-dimensional, for which multivariate observations, such as temperature, salinity, nutrients, pollutants or air pressure, are recorded across time. Biological and environmental processes are usually part of intricate networks of dynamical processes where their evolution in time is a key aspect.

Evolving, online or adaptive intelligent systems [4] are meant to be applied on sequential data or streams of data. These systems distinguish themselves from conventional offline learning methods and previous online methods in that their structure (in addition to their parameters) evolves in order to account for new data as it becomes available. Recently, there has been an increase of interest in this field. Specially during the last decade many advances have been made within the area of evolving neuro-fuzzy systems for modeling and control [4–6]. Two advantages of these methods are specially relevant for spatio-temporal biological and environmental datasets. On the one hand, they rely on simple and fast algorithms, usually operating in a one-pass manner. Thus, large datasets can be processed efficiently. On the other hand, their parameters but more importantly their structure evolve in order to accommodate for new data. Thus, large datasets can be efficiently processed online in a fully adaptive manner.

The approach proposed here is to perform kernel smoothing regression in an estimated representation of the (potentially evolving) probability distribution rather than on the whole sequence of observations. This is achieved by performing vector

* Corresponding author. Tel.: +358 9 191 57866.
E-mail addresses: federico.montesinopouzols@helsinki.fi (F. Montesino Pouzols), amaury.lendasse@aalto.fi (A. Lendasse).
URLS: http://www.helsinki.fi/bioscience/consplan/ (F. Montesino Pouzols), http://research.ics.tkk.fi/eiml/ (A. Lendasse).
[1] The work of F.M.P. was supported by a Marie Curie Intra-European Fellowship for Career Development (grant agreement PIEF-GA-2009-237450) within the European Community's Seventh Framework Programme (FP7/2007–2013).

quantization on the incoming stream. This way, a kernel smoothing regression is performed in an incrementally computed density estimation. In addition, the kernel bandwidth is adapted online. All the steps involved are incremental and the method is thus suitable for online learning. The method is simple, fast and adapts in time to evolving streams of continuous data.

The remainder of the paper is organized as follows. Section 2 describes the proposed method. The method is evaluated and compared in Section 3. Results and implications are further discussed in Section 4.

2. Proposed method

Kernel regression, also called kernel smoothing regression in order to avoid confusion with other kernel methods, is a non-parametric approach in estimating the conditional expectation of a random variable y [2,7,8]: $E(y|x) = f(x)$, where y and x are the random variables and $f(\cdot)$ is a non-parametric function. The kernel smoothing regression approach is based on the kernel density estimation. It is assumed that the model estimation has the following form: $\hat{f}(x) = y + \varepsilon$, i.e., the random variable modeled can be expressed as the sum of a deterministic, functional component and a noise component.

The Nadaraya–Watson kernel regression method for function estimation is one particular case in which the Gaussian kernel is used. If n observations of input and output pairs, (x_i, y_i), are available, the estimator of $\hat{f}(\cdot)$ for a given input observation is defined as follows:

$$\hat{f}(x_0) = \frac{\sum_{i=1}^{n} K_h(x_0, x_i) y_i}{\sum_{i=1}^{n} K_h(x_0, x_i)},$$

where h is the bandwidth or smoothing parameter, and K_h is the kernel function [9]. Some common examples of kernels are Gaussian, Epanechnikov, biweight, rectangular and triangular [9,7]. A special case is the uniform kernel, which can be considered as the kernel used in the naive density estimation method [10,11].

Vector quantization (VQ) is an unsupervised method with parallelisms with methods for clustering and learning densities such as k-means and Voronoi diagrams [2]. It is a practical and popular approach in signal processing and machine learning for lossy data compression and correction as well as density estimation, i.e., the process of deriving from observed data an estimate of an underlying probability density function. A key aspect of VQ for the purposes of this work is that it allows one to approximate the probability distribution function of a process by the distribution of prototypes or codewords. In fact, the area closer to a particular codeword than to any other is inversely proportional to the density in that region of the input domain.

The approach proposed here is to perform kernel regression in an incremental estimation of the (potentially evolving) probability distribution of the incoming data stream rather than the whole sequence of observations. This is done in two stages. First, VQ is incrementally performed on the incoming stream. Second, kernel smoothing regression for each incoming observation is computed using the codebook resulting from the first stage as an estimation of the probability distribution of the incoming data. In addition, the kernel bandwidth is adapted online.

All the steps required are incremental and the method is thus suitable for online learning. The method is simple, adapts locally to fit evolving streams of data, and is fast, with run-time complexity proportional to the number of observations and their dimensionality. An scheme of this method (KSR-VQ) is shown in Fig. 1. For the sake of brevity we will refer to it as KSR-VQ.

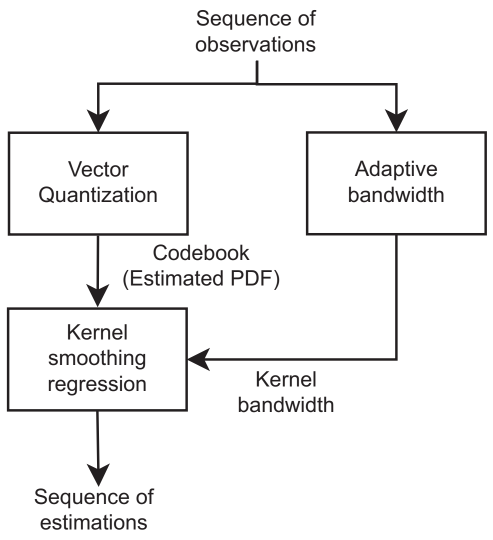


Fig. 1. Global scheme of the KSR-VQ method.

KSR-VQ takes advantage of the density matching capability of VQ. The method is defined to be fast and suitable for streams of nonstationary data and unbounded size. The two stages of KSR-VQ are detailed in what follows.

2.1. Adaptive vector quantization

The first stage of KSR-VQ is performed adaptively and in an incremental manner. Observations are processed one at a time. Let m be the current number of prototypes in the codebook, initialized to 0, and M be a maximum number of prototypes. In Algorithm 1, we show a simple version of vector quantization which is used in this paper. It should be noted that no sensitivity parameters are used.

Algorithm 1. Simple Adaptive Vector Quantization.

```
Input: Sequence of observations, $X = \{\boldsymbol{x}_i \in \mathbb{R}^d,\ i = 1, \ldots\}$
Output: Codebook, (initially empty) set of prototypes,
    $C = \{\boldsymbol{p}_j \in \mathbb{R}^d,\ j = 1, \ldots, m\},\ m \le M$
while new observations, $\boldsymbol{x}_i$, arrive do
    if $m < M$ then
        Add $\boldsymbol{x}_i$ to $C$ as a new prototype, $\boldsymbol{p}_{m+1}$;
    else
        Find in $C$ the nearest prototype $\boldsymbol{p}_{NN(i)}$ to $\boldsymbol{x}_i$
        Update the codebook with learning rate $\alpha$,
        moving $\boldsymbol{p}_{NN(i)}$ towards the sample
        point : $\boldsymbol{p}_{NN(i)} \leftarrow (1-\alpha)\boldsymbol{p}_{NN(i)} + \alpha\boldsymbol{x}_i,\quad \alpha \in [0,1]$;
```

As it will be shown in Section 3, relatively small codebooks of a few hundred prototypes can achieve satisfactory performance in a rather general setup.

2.2. Adaptive kernel smoothing regression

It is generally accepted that local adaptation of the kernel bandwidth parameter is of major importance for obtaining accurate models [7,9]. However, finding optimal or good values for the bandwidth parameter and furthermore adapting it locally is not a trivial task [12,2,7,9].