Information Sciences 320 (2015) 1–11

Contents lists available at ScienceDirect

Information Sciences

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

A mixed-integer programming approach to GRNN parameter estimation

G.E. Lee [*], A. Zaknich

School of Engineering and Information Technology, Murdoch University, WA 6150, Australia

A R T I C L E I N F O

Article history:
Received 12 December 2006
Received in revised form 20 April 2015
Accepted 25 April 2015
Available online 12 May 2015

Keywords:
General regression neural network
Least squares support vector machine
Mixed integer programming
Evolutionary computation

A B S T R A C T

A mixed-integer programming formulation for sparse general regression neural networks (GRNNs) is presented, along with a method for estimating GRNN parameters based on techniques drawn from support vector machines (SVMs) and evolutionary computation. GRNNs have been widely used for regression estimation, learning a function from a set of input/output examples, but they utilise the full set of training examples to evaluate the interpolation function. Sparse GRNNs choose a subset of the training examples, analogous to the support vectors chosen by SVMs. Experimental comparisons are made with non-sparse GRNNs and with sparse GRNNs whose centres are randomly chosen or are chosen using vector quantisation of the input domain. It is shown that the mixed-integer programming approach leads to lower prediction errors compared with previous approaches, especially when using a small fraction of the training examples.

1. Introduction

There is a family of important general regression techniques based on spherical kernels that have similar formulations to the nonparametric regression work of Nadaraya [9] and Watson [21] from 1964. These include the 1991 developments of Specht's General Regression Neural Network (GRNN) [15], and the Modified Probabilistic Neural Network (MPNN) [23] by Zaknich et al., and the earlier method of Moody and Darken [7,8] from 1988. The GRNN is essentially a direct implementation of Nadaraya and Watson's formulation using a neural network structure. Introduced independently and at the same time as the GRNN, the MPNN was spawned from a combination of Specht's Probabilistic Neural Network classifier [14] and ideas from Poggio and Girosi's work on 3-layer networks [11] and is similar to the GRNN. Both the GRNN and MPNN are based on spherical kernel functions or radial basis functions (RBFs), all having the same bandwidth, as opposed to the method of Moody and Darken which is based on spherical kernel functions each having different bandwidths. Moody and Darken's method estimates its parameters by using a clustering technique in combination with least mean squared (LMS) adaptation, whereas the GRNN and MPNN are trained more simply by adjusting their single bandwidth (or smoothing) parameter.

The main limitation of the original nonparametric formulation is that the network size is unnecessarily large because it directly incorporates all the training samples into its structure. In his original work Specht [15] mentioned that it was possible to reduce the GRNN's network size by replacing the individual samples of the nonparametric formulation with a smaller number of representative cluster centres and thereby transform it into a more efficient semiparametric form.

* Corresponding author. Tel.: +61 8 9360 6098; fax: +61 8 9360 6346.
E-mail address: gareth.lee@murdoch.edu.au (G.E. Lee).

http://dx.doi.org/10.1016/j.ins.2015.04.052
0020-0255/© 2015 Elsevier Inc. All rights reserved.

Support Vector Machines (SVMs) are a family of network architectures that can be used for classification or regression estimation (Ref. [12] provides a thorough introduction to the subject). This paper is only concerned with support vector regression (SVR) [13]. SVMs were proposed by Vapnik and Chervonenkis [20] as a realisation of their Statistical Learning Theory, or VC Theory, developed over the past four decades. SVMs have been widely used since they have the capacity to condense information in the training data to a minimal set of Support Vectors (SVs) in a similar way to the GRNN representative cluster centres, by creating a sparse solution and selecting a simplified model of the target function. One variation on the original form of SVM utilises a squared error metric and is related to ridge regression problems [16]. These Least Squares SVMs (LS-SVMs) [18,16] can be trained by solving a set of linear simultaneous equations rather than a quadratic programming problem.

The work described here aims to develop a sparse multivariate interpolation model that offers a high level of fidelity, when measured on an independent test set. A sparse model offers reduced computational burden when the model is used, albeit at the cost of greater complexity, searching for good parameters to control the model. Since the interpolation model often operates in a real-time environment, such as in a signal processing or control system, whereas the parameter estimation can occur off-line, this is a good trade-off for many applications.

This paper shows a novel mixed-integer programming (MIP) formulation for sparse GRNNs and shows how GRNN parameters can be estimated using techniques drawn from LS-SVMs and evolutionary computation. Unlike previous GRNN methods the MIP-based GRNN selects a set of support vectors, from among the training examples, specifically to minimise the mean squared error on an independent test set. Experimental comparisons are made with non-sparse GRNNs, with sparse GRNNs whose centres are randomly chosen and GRNNs with centres chosen using vector quantisation of the input domain. It will be shown that the new method performs better experimentally than SVR for very sparse models, which may be explained by the improved interpolation offered by the GRNN compared with the RBF-like interpolation provided by SVMs. This improvement is discussed in the following section.

The paper is structured as follows. Section 2 provides an introduction to GRNNs, their relationship to radial basis functions and SVMs, whereas Section 3 describes the optimisation problem associated with sparse LS-SVMs. A mixed-integer programming optimisation problem is introduced in Section 4 provides that can be used to estimate the parameters for a sparse GRNN interpolator, while Section 5 describes how this problem can be solved using evolutionary computation techniques. Section 6 presents a heuristic approach for determining the optimal kernel bandwidth which is used in Section 7 to evaluate the new method on two example problems. The conclusions drawn from the work are presented, along with likely future directions in Section 8.

2. GRNN interpolation

The GRNN provides an interpolation function f, inferred from a set of N input/output training pairs $\{(\mathbf{x}_i, y_i)\}_{i=1}^N$ that map from a multivariate domain $\mathbf{x}_i \in \mathbb{R}^p$ to a scalar range $y_i \in \mathbb{R}$. It is assumed that the training set are samples of an unknown target function that is to be estimated. The non-parametric form is defined by (1).

$$f(\mathbf{x}) = \frac{\sum_{i=1}^N y_i k(\mathbf{x}_i, \mathbf{x})}{\sum_{i=1}^N k(\mathbf{x}_i, \mathbf{x})}. \tag{1}$$

The function may be evaluated at any point $\mathbf{x} \in \mathbb{R}^p$ within the subspace spanned by the set $\{\mathbf{x}_i\}$ and provides interpolation between the discrete set of values $\{y_i\}$. No explicit bias term (b) is required in a GRNN since any bias can be directly included in the values of the y_i coefficients. In this respect a GRNN differs from most RBF-like neural network architectures, such as the SVM. For ease of comparison with the SVM equations, k is used to denote any Mercer kernel [12]. In this paper kernels will only be chosen from the subset of radial basis function kernels and in experiments Gaussian kernels will be used, as defined by (2).

$$k(\mathbf{x}_1, \mathbf{x}_2; \sigma) = \exp\left(\frac{-\|\mathbf{x}_1 - \mathbf{x}_2\|^2}{2\sigma^2}\right). \tag{2}$$

Such kernels are controlled by an additional bandwidth parameter, which in the Gaussian case is the standard deviation σ. In the case when a very small bandwidth is chosen the interpolation function $f(\mathbf{x})$ acts as a vector quantiser, by establishing a Voronoi cell around each $\mathbf{x}_i$ and setting a plateau with height y_i within this region. This leads to rapid transitions at the edges of each Voronoi cell as $f(\mathbf{x})$ switches to the heights of the neighbouring Voronoi cells. Better interpolation is achieved when a larger value is used for the kernel bandwidth σ, leading to more gradual transitions from each cell to its neighbours. In the case of Gaussian kernel functions, the transitions become smooth Sigmoidal functions.

Fig. 1 demonstrates the different form of interpolation offered by a RBF-like structure, such as a conventional SVM (shown in part (a)), as opposed to a GRNN (in part (b)). Both interpolators are modelling a single cycle of $\sin(x)$, sampled as nine (x_i, y_i) pairs and marked with crosses. It can be seen that the inclusion of a denominator term (1) leads to step-like interpolation as opposed to the superposition of kernel functions offered by RBF-like networks. In both cases the kernel bandwidth has been deliberately kept small to emphasise the differences. It can be seen that the GRNN interpolator is less sensitive to the choice of kernel bandwidth, which is particularly beneficial when a single global bandwidth is used and yet the training