Robust truncated support vector regression

Yong-Ping Zhao [a,*], Jian-Guo Sun [b]

[a] School of Mechanical Engineering, Nanjing University of Science and Technology, Nanjing 210094, China
[b] Department of Energy and Power Engineering, Nanjing University of Aeronautics and Astronautics, Nanjing 210016, China

ARTICLE INFO

ABSTRACT

In this paper, we utilize two ε-insensitive loss functions to construct a non-convex loss function. Based on this non-convex loss function, a robust truncated support vector regression (TSVR) is proposed. In order to solve the TSVR, the concave–convex procedure is used to circumvent this problem though transforming the non-convex problem to a sequence of convex ones. The TSVR owns better robustness to outliers than the classical support vector regression, which makes the TSVR gain advantages in the generalization ability and the number of support vector. Finally, the experiments on the synthetic and real-world benchmark data sets further confirm the effectiveness of our proposed TSVR.

© 2009 Elsevier Ltd. All rights reserved.

1. Introduction

The support vector machine (SVM) (Burges, 1998; Cristianini & Shawe-Taylor, 2000; Schölkopf & Smola, 2002; Vapnik, 1995), which is based on statistical theory and structural risk minimization principle, proposed by Vapnik and his group enjoys successful real-life applications such as classification and regression. As a state-of-the-art technique, SVM has better generalization ability compared with other machine learning methods like artificial neural network (ANN). However, for SVM, the computational complexity is expensive in the training stage, which needs to solve a quadratic programming problem, and its solution only depends on a small subset of the training samples, viz. the set of support vectors. For classification problem, SVM has already obtained good results, and in the regression domain, there exists a corresponding machine called support vector regression (SVR), whose training computational cost is as expensive as SVM's, i.e., $O(N^3)$, where N is the ensemble number of the training samples. To circumvent their problems, so far, many convex optimization algorithms, such as Chunking (Osuna, Freund, & Girosi, 1999), SMO (Platt, 1999; Shevade, Keerthi, Bhattacharyya, & Murthy, 2000), SVMlight (Joachims, 1999), SVMTorch (Collobert & Bengio, 2001), and software LIBSVM (Chang & Lin, 2001), have been proposed to accelerate the training speed by optimizing a small subset of the variables in the dual during the iteration procedure.

Generally, most people study the SVM from a geometrical viewpoint, i.e., SVM is a large margin based classifier for binary classification. It seeks a linear hyperplane to separate the training samples as largely as possible in the high dimensional feature space.

Thereby, the large margin helps SVM improve its generalization ability. This view is familiar to us. However, there exists another viewpoint to explain SVM, that is, it fits the regularization learning framework of *loss + penalty* with the hinge loss (Evgeniou, Pontil, & Poggio, 2000; Wahba, 1999). In the regularization learning framework, the loss function is utilized to minimize the empirical error so that the resultant model can fit the training samples well. The penalty term is used to regularize the resulting model so as to escape from overfitting. Actually, SVR can be also interpreted from the regularization learning framework. For SVR, the penalty term, which is used to control the model complexity, is the same as that of SVM, while SVR selects the ε-insensitive function to construct the loss term as a surrogate of the hinge loss for SVM. If we distinguish between SVM and SVR only from this point-of-view namely *loss + penalty*, then we can understand that the loss terms lead to their differences. Practically, apart from the hinge loss in SVM and the ε-insensitive loss in SVR, the loss functions play a significant part in supervised learning, which construct different optimization objectives so as to discriminate different learning algorithms, for example, log loss for logistic regression, exponential loss for Boosting, least squares loss for ridge regression, and so on. Overall these loss functions are all convex which own the application and theory advantages over non-convex ones, because they can be easily used, efficiently computed, and conveniently analysed. However, non-convex loss functions have superiority to convex ones in generalization performance, robustness and scalability, and they have attracted much attention in recent years. Shen, Tseng, Zhang, and Wong (2003) firstly proposed ψ-learning and indicated that non-convex loss functions yield fast convergence rates to the Bayes limit, but Steinwart and Scovel (2004) revealed that SVMs can achieve comparable convergence rates with the hinge loss under a lot of similar assumptions. Subsequently, Liu, Shen, and Doss (2005), Liu and Shen (2006) profoundly studied the use of ψ loss, and Collobert,

Sinz, Weston, and Bottou (2006) shown that the non-convexity can provide scalability advantages over convexity. Recently, Wu and Liu (2007) proposed a robust truncated hinge loss SVM, which is demonstrated to be more robust to outliers and to deliver more accuracy classifiers. However, these studies only focus on the classification domain, whereas there are few reports on regression domain. Hence, inspired by these studies, in our paper, a robust truncated support vector regression (TSVR) is proposed, which is based on a truncated ε-insensitive loss function. The TSVR cannot only enhance prediction accuracy but also curtail the number of support vectors to shorten prediction time in a way through discarding the outliers existing in the training set. Because the sophisticated convex optimization algorithms are not used to deal with the non-convex optimization problem of the TSVR, thus the concave–convex procedure (CCCP) (Yuille & Rangarajan, 2003) is employed to transform the non-convex optimization to a sequence of convex ones, which can be solved using the sophisticated convex optimization techniques. Finally, experiments on the synthetic and real-world benchmark data sets confirm the effectiveness, namely robustness to outliers, of our proposed TSVR.

In this section, we will introduce the organization of our paper. In Section 2, the classical SVR is introduced. In the following, we construct a non-convex loss function. Based on this non-convex loss function, the truncated SVR is proposed in Section 4. Section 5 gives the convergence and computational complexity of the proposed TSVR. To confirm the effectiveness, we utilize the synthetic and real-world benchmark data sets to test the TSVR in Section 6. Finally, the conclusions follow.

2. Support vector regression

Considering N pairs of training samples $\{(\boldsymbol{x}_i, d_i)\}_{i=1}^{N}$, where $\boldsymbol{x}_i \in \Re^n$ is the input variable and d_i is the corresponding target, and the ε-insensitive loss function, the SVR model can be obtained through solving the following optimization problem:

$$\min_{\boldsymbol{w},b} \quad \left\{ \frac{1}{2}\|\boldsymbol{w}\|^2 + C\sum_{i=1}^{N}\left(\xi_i + \xi_i^*\right) \right\}$$

$$\text{s.t.} \quad d_i - (\boldsymbol{w} \cdot \varphi(\boldsymbol{x}_i) + b) \leqslant \varepsilon + \xi_i^* \qquad (1)$$
$$\boldsymbol{w} \cdot \varphi(\boldsymbol{x}_i) + b - d_i \leqslant \varepsilon + \xi_i$$
$$\xi_i^*, \quad \xi_i \geqslant 0, \ i = 1,\ldots,N.$$

where C is a regulator which is used to control the tradeoff between the model complexity and training error, and $\varphi(\cdot)$ is usually a non-linear mapping which is induced by a kernel function. In order to solve (1), we need to construct a Lagrange function, and from the Karush–Kuhn–Tucker (KKT) conditions, we can get the following dual optimization problem:

$$\min_{\boldsymbol{\alpha}^*,\boldsymbol{\alpha}} \quad \frac{1}{2}\sum_{i,j=1}^{N}\left(\alpha_i^* - \alpha_i\right)\left(\alpha_j^* - \alpha_j\right)k(\boldsymbol{x}_i,\boldsymbol{x}_j)$$

$$+ \varepsilon\sum_{i=1}^{N}\left(\alpha_i^* + \alpha_i\right) - \sum_{i=1}^{N}d_i(\alpha_i^* - \alpha_i) \qquad (2)$$

$$\text{s.t.} \quad \sum_{i=1}^{N}\left(\alpha_i^* - \alpha_i\right) = 0,$$
$$0 \leqslant \alpha_i^* \leqslant C, \quad 0 \leqslant \alpha_i \leqslant C, \ i = 1,\ldots,N,$$

where $k(\cdot,\cdot)$ is a kernel function which represents the inner product between two vectors in the high dimensional (even infinite) feature space. Usually, $k(\cdot,\cdot)$ can be chosen as Gaussian, polynomial, or MLP. After solving (2), the following predictor, viz. SVR, can be got

$$f(\boldsymbol{x}) = \sum_{\boldsymbol{x}_i \in SV}\left(\alpha_i^* - \alpha_i\right)k(\boldsymbol{x},\boldsymbol{x}_i) + b, \qquad (3)$$

where SV is the support vectors set.

3. The non-convex loss function

According to the principle of *loss + penalty*, let $\varepsilon = \varepsilon_1$, and then (1) can be rewritten as

$$\min_{\boldsymbol{w},b} \quad \left\{ g = \frac{1}{2}\|\boldsymbol{w}\|^2 + C\sum_{i=1}^{N}H_{\varepsilon_1}(|z_i|) \right\}, \qquad (4)$$

where $H_{\varepsilon_1}(\cdot)$ is a ε_1-insensitive loss function which is illustrated as Fig. 1, and the ε_1-insensitive loss function can be written as $H_{\varepsilon_1}(|z_i|) = \max\{0, |z_i| - \varepsilon_1\}$ with $z_i = d_i - \boldsymbol{w} \cdot \varphi(\boldsymbol{x}_i) - b = d_i - f(\boldsymbol{x}_i)$. As we know, the samples with the losses $|z_i| \geqslant \varepsilon_1$ are regarded as support vectors, whereas the others are not. As for this point, we can give an intuitive explanation. Given that the ε_1-insensitive loss function is differentiable with a smooth approximation to the small interval $|z_i| \in [\varepsilon_1 - \delta, \varepsilon_1 + \delta]$ near the insensitive point, where δ is a small positive constant. Let $\frac{\partial g}{\partial \boldsymbol{w}} = 0$, and then we have

$$\boldsymbol{w} = -C\sum_{i=1}^{N}sign(z_i)H'_{\varepsilon_1}(|z_i|)\varphi(\boldsymbol{x}_i), \qquad (5)$$

where $sign(z_i) = \begin{cases} 1 & \text{if } z_i \geqslant 0 \\ -1 & \text{if } z_i < 0 \end{cases}$. When samples with $|z_i| < \varepsilon_1$ lie in the flat area, they cannot become support vectors because $H'_{\varepsilon_1}(|z_i|)$ is equal to 0. Similarly, samples with $|z_i| \geqslant \varepsilon_1$ can become support vectors because of their corresponding derivations being 1. Generally, outliers own larger $|z_i|s$ than the normal samples. That's to say, outliers are usually viewed as support vectors owing to their $|z_i|s > \varepsilon_1$, which are unreasonable and naturally degrade the generalization performance of SVR. Thereby, we must take actions to discard outliers as support vectors. From the aforementioned fact, we know that when the samples lie in the flat area, they cannot become support vectors. In addition, outliers usually introduce larger loss $|z_i|s$ than the normal samples. Hence, we only need to construct a loss function which lets outliers lie in the flat area. Following this line, we combine two ε-insensitive loss functions to form a loss function (illustrated as Fig. 3) as

$$R(|z_i|) = H_{\varepsilon_1}(|z_i|) - H_{\varepsilon_2}(|z_i|), \qquad (6)$$

where $H_{\varepsilon_2}(|z_i|)$ is also a ε-insensitive loss function (see Fig. 2) with $\varepsilon_2 > \varepsilon_1$. If an appropriate ε_2 is selected, then we can let that outliers with $|z_i| \geqslant \varepsilon_2$ locate in the flat area so that they do not become support vectors, and through discarding outliers, we can improve the generalization performance, namely prediction accuracy, of SVR and curtail the number of support vectors. Visually, we can call $R(|z_i|)$ the truncated ε-insensitive loss function. If $H_{\varepsilon_1}(|z_i|)$ in (4) is
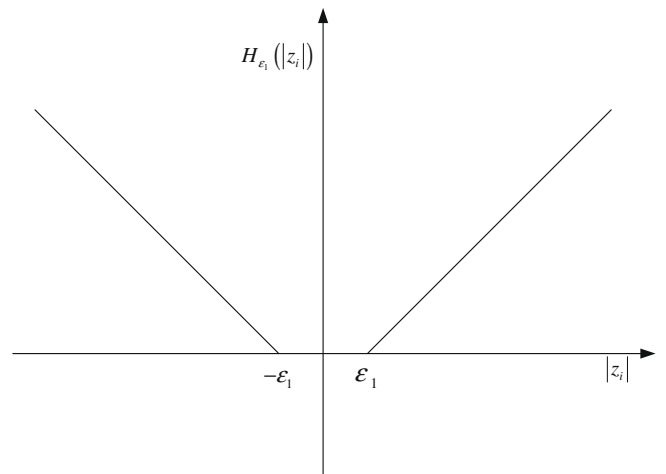

Fig. 1. ε_1-Insensitive loss function.