A Unified Loss Function in Bayesian Framework for Support Vector Regression

Wei Chu ENGP9354@NUS.EDU.SG
S. Sathiya Keerthi MPESSK@NUS.EDU.SG
Chong Jin Ong MPEONGCJ@NUS.EDU.SG
Control Division, Department of Mechanical Engineering, National University of Singapore, 10 Kent Ridge Crescent, Singapore, 119260

Abstract

In this paper, we propose a unified non-quadratic loss function for regression known as soft insensitive loss function (SILF). SILF is a flexible model and possesses most of the desirable characteristics of popular non-quadratic loss functions, such as Laplacian, Huber's and Vapnik's ε-insensitive loss function. We describe the properties of SILF and illustrate our assumption on the underlying noise model in detail. Moreover, the introduction of SILF in regression makes it possible to apply Bayesian techniques on Support Vector methods. Experimental results on simulated and real-world datasets indicate the feasibility of the approach.

1. Introduction

In regression problems, different choices of loss functions arise from various assumptions about the distribution of the noise in measurement. The most common loss function is the quadratic function corresponding to a Gaussian noise model with zero mean, and a standard deviation that does not depend on the inputs. The Gaussian loss function is used popularly as it has nice analytical properties. However, one of the potential difficulties of the standard quadratic loss function is that it receives the large contributions from outliers that have particularly large errors. If there are long tails on the distributions then the solution can be dominated by a very small number of outliers. Techniques that attempt to solve this problem are referred to as robust statistics (Huber, 1972). Several non-quadratic loss functions have been introduced to reduce the sensitivity to the outliers, such as the Laplacian loss function and the Huber's loss function.

The ε-insensitive loss function (ILF) was proposed by Vapnik in Support Vector machines (SVM) for regression (SVR) (Vapnik, 1995). SVR involves the solution of a convex quadratic programming optimization problem. The advantages of SVR are: global minimum solution, fast training speed and sparseness in sample selection. The performance of SVR crucially depends on the shape of the kernel function and other hyperparameters that represent the characteristics of the noise distributions in the training data. Typically, Bayesian methods are regarded as suitable tools to determine these hyperparameters. However, these methods are difficult to use in SVR due to the lack of smoothness of ILF.

In the literature on Bayesian interpretations of Support Vector classifier, Platt (1999) gave a probabilistic explanation on the outputs, Kwok (1999) built up an evidence framework, and Sollich (2001) proposed Bayesian methods with evidence and error bar. Tipping (2000) put forward relevance vector machine, an alternative Bayesian treatment to SVM. In this paper, we propose a unified framework for popular non-quadratic regression loss functions. This is done using a novel loss function known as soft insensitive loss function (SILF). SILF possesses most of the virtues in popular non-quadratic loss functions such as insensitivity to the outliers, differentiability and sparseness in sample selection etc. In addition, the introduction of SILF in regression makes it possible to apply Bayesian techniques on Support Vector methods, while preserving their individual advantages.

The paper is organized as follows: we first review the popular non-quadratic loss functions and put forward the SILF and its properties in section 2; then we introduce the SILF into likelihood function in the Bayesian framework to tackle regression problems, and discuss hyperparameter inference in section 3; and we show the results of numerical experiments to verify the approach in section 4.

2. A Unified Non-quadratic Loss Function

2.1 Non-quadratic Loss Functions

Suppose that a set of training data D has been collected by randomly sampling a function f, defined on R^d. As the measurements are always corrupted by additive noise, each training sample can be represented as

$$y_i = f(x_i) + \delta_i \qquad i = 1,\ldots,n \tag{1}$$

where the δ_i are independent random variables with a given distribution. Our goal is to infer the function f, or an

estimate of it, from the finite data set *D*. In the Bayesian approach, we regard that the function *f* as the realization of a random field with a known prior probability. Then the posterior probability of *f* given the data *D* can be written as follows:

$$P(f \mid D) \propto P(D \mid f)P(f) \tag{2}$$

where $P(D \mid f)$ is the conditional probability of the data *D* given the function *f* and $P(f)$ is the priori probability of the random field *f*. The probability $P(D \mid f)$, also known as likelihood, is essentially a model of the noise, and, if the additive noise δ_i is i.i.d. random variable with probability distribution $P(\delta_i)$, it can be written as:

$$P(D \mid f) = \prod_{i=1}^{n} P(y_i - f(x_i) \mid f) = \prod_{i=1}^{n} P(\delta_i) \tag{3}$$

Furthermore, we usually assume that these noise random variables possess identical probability distribution, and that the conditional distribution $P(y - f(x) \mid f)$ can be written in the form

$$P(y - f(x) \mid f) \propto \exp(-C \cdot \ell(\delta)) \tag{4}$$

where $\ell(\cdot)$ is called the loss function and *C* is a parameter greater than zero. Thus, our assumption about the noise distribution will completely determine the form of the loss function.

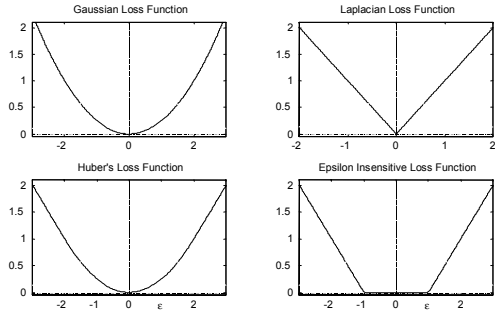

Figure 1. Graphs of non-quadratic loss functions along with Gaussian loss function.

LAPLACIAN LOSS FUNCTION

If we assume that the additive noise is Laplacian noise, then the loss function is

$$\ell(\delta) = |\delta| \tag{5}$$

which is also called as the L_1 loss function.

HUBER'S LOSS FUNCTION

Huber's loss function (Huber, 1972) is defined as

$$\ell(\delta) = \begin{cases} \frac{1}{4\varepsilon}\delta^2 & \text{if } |\delta| \le 2\varepsilon \\ |\delta| - \varepsilon & \text{otherwise} \end{cases} \tag{6}$$

where $\varepsilon > 0$.

ε-INSENSITIVE LOSS FUNCTION

The ε-insensitive loss function (ILF) was introduced by Vapnik (1995), which is defined as

$$\ell(\delta) = \begin{cases} 0 & \text{if } |\delta| \le \varepsilon \\ |\delta| - \varepsilon & \text{otherwise} \end{cases} \tag{7}$$

where $\varepsilon > 0$.

From their definitions and Figure 1, we notice that Huber's loss function and ε-insensitive loss function become the Laplacian loss function when ε=0. Huber's loss function can be thought of as a mixture between Gaussian and Laplacian loss functions. Laplacian and ε-insensitive loss functions are non-smooth. Based on these observations, we blend their features together and propose a novel loss function, namely soft insensitive loss function (SILF).

2.2 Soft Insensitive Loss Function

The SILF can be seen as a unified loss function of the above-mentioned non-quadratic loss functions. It is defined as:

$$\ell_{\varepsilon,\beta}(\delta) = \begin{cases} -\delta - \varepsilon & \text{if } \delta \in \Delta_{C^*} = (-\infty, -(1+\beta)\varepsilon) \\ \frac{(\delta + (1-\beta)\varepsilon)^2}{4\beta\varepsilon} & \text{if } \delta \in \Delta_{M^*} = [-(1+\beta)\varepsilon, -(1-\beta)\varepsilon] \\ 0 & \text{if } \delta \in \Delta_0 = (-(1-\beta)\varepsilon, (1-\beta)\varepsilon) \\ \frac{(\delta - (1-\beta)\varepsilon)^2}{4\beta\varepsilon} & \text{if } \delta \in \Delta_M = [(1-\beta)\varepsilon, (1+\beta)\varepsilon] \\ \delta - \varepsilon & \text{if } \delta \in \Delta_C = ((1+\beta)\varepsilon, +\infty) \end{cases}$$

where $0 < \beta \le 1$, $\varepsilon > 0$. (8)

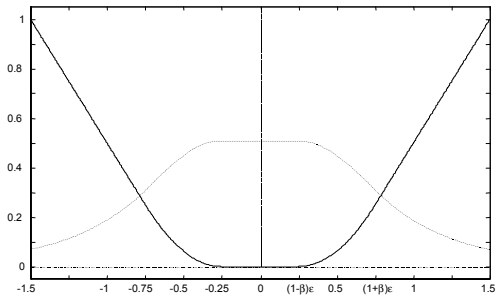

Figure 2. Graph of soft insensitive loss function (solid line) and corresponding noise density function (dot line). β=0.5, ε=0.5 and C=2 in noise model.

The properties of SILF are entirely controlled by two parameters, β and ε. For a fixed ε, SILF approaches the ε-insensitive loss function as $\beta \to 0$; on the other hand, as $\beta \to 1$, it approaches the Huber's robust loss function. In addition, SILF becomes the Laplacian loss function as $\varepsilon \to 0$.

2.2.1 DERIVATIVES

Derivatives of the loss function are needed in Bayesian methods. The first order derivative of SILF can be written as

$$\frac{\partial \ell_{\varepsilon,\beta}(\delta)}{\partial \delta} = \begin{cases} -1 & \text{if } \delta \in \Delta_{C^*} \\ \frac{\delta + (1-\beta)\varepsilon}{2\beta\varepsilon} & \text{if } \delta \in \Delta_{M^*} \\ 0 & \text{if } \delta \in \Delta_0 \\ \frac{\delta - (1-\beta)\varepsilon}{2\beta\varepsilon} & \text{if } \delta \in \Delta_M \\ 1 & \text{if } \delta \in \Delta_C \end{cases} \tag{9}$$

where $0 < \beta \le 1$, $\varepsilon > 0$.

It is easy to see that the function is not twice differentiable, but the second order derivative exists almost everywhere:

$$\frac{\partial^2 \ell_{\varepsilon,\beta}(\delta)}{\partial \delta^2} = \begin{cases} \frac{1}{2\beta\varepsilon} & if \quad \delta \in \Delta_{M^*} \cup \Delta_M \\ 0 & otherwise \end{cases} \tag{10}$$

where $0 < \beta \le 1$, $\varepsilon > 0$.

2.2.2 Noise Density Function

The density function of the additive noise in measurement corresponding to the choice of SILF is

$$p(\delta) = \frac{1}{Z_D} \exp\left(-C \cdot \ell_{\varepsilon,\beta}(\delta)\right) \tag{11}$$

where $Z_D = \int \exp\left(-C \cdot \ell_{\varepsilon,\beta}(\delta)\right) d\delta$

It is possible to evaluate the integral and write Z_D as:

$$Z_D = 2(1-\beta)\varepsilon + 2\sqrt{\frac{\beta\varepsilon\pi}{C}} erf\left(\sqrt{C\beta\varepsilon}\right) + \frac{2}{C}\exp(-C\beta\varepsilon) \tag{12}$$

where $erf(x) = \frac{2}{\sqrt{\pi}} \int_0^x \exp\left(-t^2\right) dt$

2.3 A General Noise Model

Pontil, Mukherjee & Girosi (1998) has shown that the use of Vapnik's ε-insensitive loss function is equivalent to a model of additive and Gaussian noise, where the variance and mean of the Gaussian are i.i.d. random variables. We now derive the noise model corresponding to SILF we proposed.

If the uncertainties in measurement conditions are taken into account, it seems reasonable to discard the popular assumption that noise variables δ_i are identically distributed. Moreover, we assume that the noise variables δ_i have probability distributions P_i which are Gaussians, but do not necessarily have zero means and identical variances. Thus, the noise distributions P_i can be assumed to be

$$P_i(\delta_i) = \frac{1}{\sqrt{2\pi}\sigma_i} \exp\left(-\frac{(\delta_i - t_i)^2}{2\sigma_i^2}\right) \tag{13}$$

where σ_i denotes standard deviation of the ith sample and t_i denotes mean of the ith sample.

Here, we allow for the fact that the noise could be biased in this model, and consider σ_i and t_i as i.i.d. random variables to model the uncertainties in measurement. Therefore, $P_i(\delta_i)$ can be interpreted as $P_i(\delta_i|\sigma_i,t_i)$, the conditional probability of δ_i given σ_i and t_i . Then we compute the marginal of the likelihood (3), integrating over $\sigma = \{\sigma_1,\ldots,\sigma_n\}$ and $t = \{t_1,\ldots,t_n\}$:

$$P(D \mid f) = \prod_{i=1}^{n} P_i(\delta_i) = \int d\sigma \int dt \prod_{i=1}^{n} P_i(\delta_i \mid \sigma_i, t_i) P(\sigma, t) \tag{14}$$

On the assumption that σ and t are i.i.d. random variables, we obtain that

$$P(\sigma,t) = \prod_{i=1}^{n} P(\sigma_i,t_i) = \prod_{i=1}^{n} \mu(\sigma_i)\lambda(t_i) \tag{15}$$

where $\mu(\cdot)$ and $\lambda(\cdot)$ denote the density distribution of σ and t respectively. Finally the likelihood (3) can be given as

$$P(D \mid f) = \prod_{i=1}^{n} \int d\sigma_i \int dt_i P(\delta_i \mid \sigma_i, t_i)\lambda(t_i)\mu(\sigma_i) \tag{16}$$

where $P(\delta_i \mid \sigma_i, t_i)$ is defined as (13).

Suppose that we choose SILF as loss function, then from (3) and (11), the likelihood can also be written as:

$$P(D \mid f) = \frac{1}{(Z_D)^n} \exp\left(-C \sum_{i=1}^{n} \ell_{\varepsilon,\beta}(y_i - f(x_i))\right) \tag{17}$$

where $\ell_{\varepsilon,\beta}(\cdot)$ is defined in (8).

Since (16) and (17) are equal, we can easily see that the loss function can also be expressed in the form as follow:

$$\ell_{\varepsilon,\beta}(\delta) = -\frac{1}{C}\log \int_0^{\infty} d\sigma \int_{-\infty}^{+\infty} dt \frac{Z_D}{\sqrt{2\pi\sigma}} \exp\left(-\frac{(\delta - t)^2}{2\sigma^2}\right)\lambda(t)\mu(\sigma) \tag{18}$$

where we drop off the subscript i in σ_i and t_i, as they are identical random variables. Thus, using a loss function with an integral representation of the form (18) is equivalent to assuming that the noise is Gaussian, but the mean and the variance of the noise are random variables with individual probability density function.

2.3.1 Density Function of Standard Deviation

We derive the density function for σ in the following. By rearranging the integral in (18), we obtain:

$$e^{-C\ell_{\varepsilon,\beta}(\delta)} = \int_{-\infty}^{+\infty} dt\ \lambda(t) G(\delta - t) \tag{19}$$

where $G(x) = \int_0^{\infty} d\sigma \frac{Z_D}{\sqrt{2\pi\sigma}} \mu(\sigma) \exp\left(-\frac{x^2}{2\sigma^2}\right)$ (20)

We notice that the function SILF becomes the Laplacian loss function (5) when $\varepsilon = 0$. In this case, the noise distribution becomes Laplacian distribution:

$$P(\delta) = \frac{1}{Z_D}\exp(-C|\delta|) \tag{21}$$

where $Z_D = 2/C$. It is also known that this Laplacian distribution is an unbiased noise distribution, i.e. the density function of its mean t is a delta function at zero (Pontil et al., 1998). Using this fact and the expression in (20), (19) can be simplified as: [1]

$$e^{-C|\delta|} = G(\delta) = \int_0^{\infty} \sqrt{\frac{2}{\pi}} \frac{1}{\sigma C} \mu(\sigma) \exp\left(-\frac{\delta^2}{2\sigma^2}\right) d\sigma \tag{22}$$

Since $2\sqrt{\pi} e^{-\sqrt{x}} = \int_0^{\infty} \eta^{-\frac{3}{2}} \cdot \exp\left(-\frac{1}{4\eta}\right) \cdot \exp(-x\eta) d\eta$ (23)

holds for any x, it follows that by setting $2C^2\sigma^2 = 1/\eta$ and $x = C^2\delta^2$, we get

$$e^{-C|\delta|} = \int_0^{\infty} \sqrt{\frac{2}{\pi}} C \exp\left(-\frac{C^2\sigma^2}{2}\right) \exp\left(-\frac{\delta^2}{2\sigma^2}\right) d\sigma \tag{24}$$

Comparing (22) with (24), we find that the density function of the standard deviation is a ***Rayleigh*** distribution of the form:

$$\mu(\sigma) = C^2\sigma \exp\left(-\frac{C^2\sigma^2}{2}\right) \tag{25}$$

[1] Such a class of loss function defined by (22) is an extension of the model discussed by Girosi (1991).

2.3.2 DENSITY DISTRIBUTION OF MEAN

So far, we know $G(\delta)=\exp(-C\cdot|\delta|)$ from (22) and $\lambda(t)$ is a delta function at zero when $\varepsilon = 0$. It remains to obtain the explicit expression of $\lambda(t)$ for $\varepsilon > 0$.

Taking Fourier transformation on (19), we get

$$\widetilde{F}\left[\exp\left(-C\ell_{\varepsilon,\beta}(\delta)\right)\right]=\widetilde{\lambda}(\omega)\widetilde{G}(\omega) \tag{26}$$

From (22), we get $\widetilde{G}(\omega)=2C/\left(C^2+\omega^2\right)$. Thus, the Fourier transformation of $\lambda(t)$ shall be

$$\widetilde{\lambda}(\omega)=\frac{1}{2C}\left(C^2+\omega^2\right)\widetilde{F}\left[\exp\left(-C\ell_{\varepsilon,\beta}(\delta)\right)\right] \tag{27}$$

Using the differential property of $\widetilde{F}\left(f^{(n)}(t)\right)=(i\omega)^n\widetilde{F}(\omega)$, we find that $\lambda(t)$ is:

$$\lambda(t)=\frac{1}{2C}\left[C^2e^{-C\ell_{\varepsilon,\beta}(t)}-\frac{\partial^2}{\partial t^2}\left(e^{-C\ell_{\varepsilon,\beta}(t)}\right)\right] \tag{28}$$

i.e.

$$\lambda(t)=\frac{1}{2}\left[C+\frac{\partial^2\ell_{\varepsilon,\beta}(t)}{\partial t^2}-C\left(\frac{\partial\ell_{\varepsilon,\beta}(t)}{\partial t}\right)^2\right]e^{-C\ell_{\varepsilon,\beta}(t)} \tag{29}$$

From the definitions (8) (9) and (10), the density distribution of mean $\lambda(t)$ can be written in explicit form without normalization as follows:

$$\lambda(t)\propto\begin{cases}1 & \text{if} \quad t\in\Delta_0\\ \left[1+\frac{1}{2C\beta\varepsilon}-\left(\frac{|t|-(1-\beta)\varepsilon}{2\beta\varepsilon}\right)^2\right]e^{-C\ell_{\varepsilon,\beta}(|t|)} & \text{if} \quad t\in\Delta_{M^*}\cup\Delta_M\\ 0 & \text{if} \quad t\in\Delta_{C^*}\cup\Delta_C\end{cases} \tag{30}$$

Finally notice that for the class of loss function, SILF, the noise distribution (19) can be written as the convolution between the distribution of the mean $\lambda(t)$ (30) and the Laplacian distribution (22):

$$P(\delta)=\int_{-(1+\beta)\varepsilon}^{(1+\beta)\varepsilon}\lambda(t)\exp\left(-C|\delta-t|\right)dt \tag{31}$$

Equation (31) establishes a representation of the noise distribution $P(\delta)$ as a continuous superposition of Laplacian functions in the interval $[-(1+\beta)\varepsilon,(1+\beta)\varepsilon]$.

2.4 Discussion

In summary, the noise model for SILF can be interpreted as Gaussian, but it could be biased, where the standard deviation and the mean of the Gaussian are i.i.d. random variables with specific probability distributions stated in (25) and (30) respectively. Parameter C determines the distribution of standard deviation entirely, while parameter ε controls main properties of the distribution of the mean. Huber's loss function and ILF can be regarded as special cases of SILF that is wholly controlled by parameter β only. All the three loss functions share the same distribution of standard deviation σ in their noise model, but the distributions of the mean t are different. For a fixed ε, as $\beta\to 0$, Δ_{M^*} and Δ_M shrink to points $\pm\varepsilon$, and the distribution becomes delta function at $\pm\varepsilon$. Thus, the distribution of mean for ILF can be stated as uniform in the interval $(-\varepsilon,\varepsilon)$ and with two delta functions at $\pm\varepsilon$. On the contrary, as $\beta\to 1$, Δ_0 shrinks. When $\beta=1$, the distribution of mean for Huber's loss function can be stated as

$$\lambda(t)\propto\left(1+\frac{1}{2C\varepsilon}-\left(\frac{t}{2\varepsilon}\right)^2\right)e^{-\frac{Ct^2}{4\varepsilon}}\qquad t\in[-2\varepsilon,2\varepsilon] \tag{32}$$

We find that the conclusions are consistent with the results given by Pontil et al. (1998). Thus, the general noise model is an extension of the noise model discussed by Pontil et al.

3. Bayesian Inference

We introduce the soft insensitive loss function (SILF) into Bayesian methods to tackle regression problems with the purpose of integrating Support Vector methods and standard Bayesian inference in a unified probabilistic framework, while leaving as much as possible of their individual advantages intact.

3.1 Bayesian Framework

The Bayesian approach for neural networks specifies a hierarchical model with a prior distribution over hyperparameters, and specifies a prior distribution of the weights relative to the hyperparameters. Then a posterior distribution over the weights and hyperparameters will be induced for a given dataset. Neal observed that a Gaussian prior for hidden-to-output weights results in a Gaussian process prior for functions as the number of hidden units goes to infinity (Neal, 1996). Inspired by Neal's work, recent work (Williams & Rasmussen, 1996) has extended the use of Gaussian process prior to higher dimensional problems that have been traditionally tackled with other techniques, such as neural networks, decision trees etc, and good results have been obtained.

We propose Bayesian framework with a Gaussian process prior on functions as follows. We assume that the collection of training data is the realization of random variables $f(x_i)$ in Gaussian process indexed by x_i. The Gaussian process can be specified by giving covariance matrix for any finite set of variables $\{f(x_i)\}$ after normalization. Then, the covariance between the outputs corresponding to inputs x and x' is defined as:

$$E[f(x)f(x')]=\exp\left(-\frac{1}{2}\sum_{l=1}^{d}w_l\left(x_l-x'_l\right)^2\right) \tag{33}$$

where d is the dimensionality of input space, x_l denotes the lth entry of the input vector x, and w_l determines the relevance of the lth input dimension to the prediction of the output variables, which is usually called Automatic Relevance Determination (ARD) parameter (Mackay, 1993; Neal, 1996). Note that (33) is also a particular form of the Gaussian kernel in SVR. In addition, let us suppose that the additive noise in training data possesses the distribution given in (11), i.e., we choose the soft insensitive loss function (SILF) to evaluate the likelihood. Let us collect the ARD parameters w together with the parameters, C and ε, as θ, the *hyperparameter* vector. For

the given model, the posterior probability of f can be derived by Bayes' theorem:

$$P(f \mid D,\theta)=\frac{P(D \mid f,\theta)P(f \mid \theta)}{P(D \mid \theta)} \tag{34}$$

where $P(D \mid f,\theta)$ is the likelihood function of the data set D, and $P(f \mid \theta)$ is a priori probability of the Gaussian processes f.

3.1.1 A PRIORI PROBABILITY

$P(f \mid \theta)$ is a multivariate Gaussian with zero mean and covariance matrix as follows:

$$P(f \mid \theta)=\frac{1}{Z_f}\exp\left(-\frac{1}{2}f^T\Sigma^{-1}f\right) \tag{35}$$

where $f=\begin{bmatrix}f(x_1) & f(x_2) & \cdots & f(x_n)\end{bmatrix}^T$, $Z_f=(2\pi)^{n/2}\sqrt{\det(\Sigma)}$ and Σ is the $n\times n$ covariance matrix whose ijth entry is defined in (33).

3.1.2 LIKELIHOOD FUNCTION

We assume that the training data are independently drawn. The likelihood function for the given dataset D can then be written as

$$P(D \mid f,\theta)=\frac{1}{(Z_D)^n}\exp\left(-C\sum_{i=1}^{n}\ell_{\varepsilon,\beta}(y_i-f(x_i))\right) \tag{36}$$

where Z_D is defined in (11).

3.1.3 A POSTERIORI PROBABILITY

The normalizing constant $P(D \mid \theta)$ in (34) is irrelevant to the inference of the functions f, but it becomes important in hyperparameter inference, and it is known as the evidence of θ for the model. Based on (35), (36) and Bayes' theorem, the posterior probability of f can be written as:

$$P(f \mid D,\theta)=\frac{1}{Z_S}\exp\left(-C\sum_{i=1}^{n}\ell_{\varepsilon,\beta}(y_i-f(x_i))-\frac{1}{2}f^T\Sigma^{-1}f\right) \tag{37}$$

where $Z_S=\int\exp\left[-C\sum_{i=1}^{n}\ell_{\varepsilon,\beta}(y_i-f(x_i))-\frac{1}{2}f^T\Sigma^{-1}f\right]df$

Let us define

$$S(f)=C\sum_{i=1}^{n}\ell_{\varepsilon,\beta}(y_i-f(x_i))+\frac{1}{2}f^T\Sigma^{-1}f \tag{38}$$

Since $P(f \mid D,\theta)\propto\exp(-S(f))$, Maximum A Posteriori (*MAP*) estimate of f is therefore the minimizer of the following optimization problem:

$$\min_f\ S(f)=C\sum_{i=1}^{n}\ell_{\varepsilon,\beta}(y_i-f(x_i))+\frac{1}{2}f^T\Sigma^{-1}f \tag{39}$$

3.2 Support Vector Methods

For a given θ and data set D, we infer what the most probable value f_{MP} might be in (37) by minimizing the functional $S(f)$. To solve the optimization problem (39), we introduce slack variables $\xi_i=y_i-f(x_i)-(1-\beta)\varepsilon$ and $\xi_i^*=f(x_i)-y_i-(1-\beta)\varepsilon$, and get an equivalent optimization problem, which is referred to as the *primal* problem:

$$\min_{f,\xi,\xi^*}\ C\sum_{i=1}^{n}\left(c(\xi_i)+c(\xi_i^*)\right)+\frac{1}{2}f^T\Sigma^{-1}f \tag{40}$$

subject to

$$\begin{cases}y_i-f(x_i)\le(1-\beta)\varepsilon+\xi_i\\ f(x_i)-y_i\le(1-\beta)\varepsilon+\xi_i^*\\ \xi_i,\xi_i^*\ge 0\end{cases}$$

where

$$c(\xi_i^{(*)})=\begin{cases}\dfrac{(\xi_i^{(*)})^2}{4\beta\varepsilon} & if\quad \xi_i^{(*)}\in[0,2\beta\varepsilon]\\ |\xi_i^{(*)}|-\beta\varepsilon & if\quad \xi_i^{(*)}\in(2\beta\varepsilon,+\infty)\end{cases}$$

The primal problem can be solved through the technique of Lagrange multipliers, which yields the following *dual* problem:

$$\min_{\alpha,\alpha^*}\ W(\alpha,\alpha^*)=\frac{1}{2}(\alpha-\alpha^*)^T\Sigma(\alpha-\alpha^*)-\sum_{i=1}^{n}(\alpha_i-\alpha_i^*)y_i+\sum_{i=1}^{n}(\alpha_i+\alpha_i^*)(1-\beta)\varepsilon+\frac{\beta\varepsilon}{C}\sum_{i=1}^{n}(\alpha_i^{\ 2}+\alpha_i^{*2}) \tag{41}$$

subject to $\alpha^{(*)}\in[0,C]$

The optimization algorithms to solve the standard SVR can be adopted to solve the dual problem easily. For more details, please refer to (Chu, 2001), and the optimal solution f_{MP} is of the form $f_{MP}=\Sigma\cdot(\alpha-\alpha^*)$.

3.3 Hyperparameter Inference

The optimal values of hyperparameters θ can be inferred by maximizing the posterior probability $P(\theta \mid D)$ in the second level of inference. Under a given model, the posterior probability $P(\theta \mid D)$ is

$$P(\theta \mid D)=\frac{P(D \mid \theta)P(\theta)}{P(D)} \tag{42}$$

A priori on the hyperparameters $P(\theta)$ is required here. As we typically have little idea about the suitable values of θ before training data are available, we assume a flat distribution for $P(\theta)$, i.e., $P(\theta)$ is greatly insensitive to the values of θ. Therefore, the evidence $P(D \mid \theta)$ can be used to assign a preference to alternative values of the hyperparameters θ.

The evidence can be calculated by an explicit formula after using a Laplacian approximation at f_{MP}, and then hyperparameter inference may be done by gradient-based optimization methods. Whether this approximation is good or not will depend on the problem we are solving. Maximization of the evidence is only useful if such a Laplacian approximation gives a good summary of the distribution. In other cases, Monte Carlo algorithm can be applied to tackle the integral instead of evidence approximation.

We can get the evidence by an integral over all f:

$$P(D \mid \theta)=\int P(D \mid f,\theta)P(f \mid \theta)df \tag{43}$$

Using the definitions for the prior probability and likelihood in (35), (36) and $S(f)$ in (38), the evidence can be written as

$$P(D \mid \theta)=Z_f^{-1}(Z_D)^{-n}\int\exp[-S(f)]df \tag{44}$$

The marginalization can be calculated analytically by considering the Taylor expansion of $S(f)$ around its minimum $S(f_{MP})$, and retaining terms up to second order. Since the first order derivative with respect with f at the most probable point f_{MP} is zero, $S(f)$ can also be written as

$$S(f) \cong S(f_{MP}) + \frac{1}{2}(f - f_{MP})^T \cdot \left.\frac{\partial^2 S(f)}{\partial f \partial f^T}\right|_{f_{MP}} \cdot (f - f_{MP}) \quad (45)$$

where $\frac{\partial^2 S(f)}{\partial f \partial f^T} = \Sigma^{-1} + C \cdot \Lambda$

and Λ is a diagonal matrix whose iith entry is $1/(2\beta\varepsilon)$ if $y_i - f(x_i) \in \Delta_{M^*} \cup \Delta_M$, otherwise zero.

In calculating the integral, we notice that, due to the sparseness of Λ only a sub-matrix of Σ plays role in the determinant $|I + C\Sigma\Lambda|$.[2] Let us denote Σ_M as the sub-matrix of Σ obtained by deleting all the rows and columns corresponding to points $y_i - f(x_i) \notin \Delta_{M^*} \cup \Delta_M$. Then the negative log of the evidence (44) can be written as

$$-\ln P(D \mid \theta) = S(f_{MP}) + \frac{1}{2}\ln\left|I + \frac{C}{2\beta\varepsilon} \cdot \Sigma_M\right| + n \ln Z_D \quad (46)$$

where Z_D is defined in (11).

Using (46), the gradients of the negative log of the evidence, $-\ln P(D \mid \theta)$ with respect to the hyperparameters can be easily derived (Chu, 2001). Then the standard gradient-based optimization methods, such as BFGS algorithm, can be applied to update these hyperparameters towards a minimum of $-\ln P(D \mid \theta)$. Notice that the inversion of the sparse matrix Σ_M is required in calculating gradients with respect to kernel parameters in an iterative mode for hyperparameter inference. In large-scale learning tasks, the size of the matrix could be large, and then the computation of the matrix inverse will be the most time-consuming step. Since β is the pivotal parameter to determine the size of Σ_M in training, we adjust β separately, just to prevent the size of Σ_M from heavily shrinking but fix its value while updating other hyperparameters.

4. Numerical Experiments

We start this section with a simple example of regression estimation task where regression function is defined by one-dimensional *sinc* function. Then we consider estimating multidimensional regression functions for robot arm problem and a real-life problem, Boston housing.

The initial values for the hyperparameters are randomly generated in a certain region. Since we do not have enough knowledge about which dimensions in inputs are relevant to the prediction of output variables, we assume that the relevance is equal. To prevent the scalar in input dimensions from impairing this assumption, we normalize the input variables, and then set the same initial value for all ARD parameters w. This value is randomly chosen in the interval [0.5,1]. The noise information is unavailable either, so we randomly choose initial values from [0.5,2] for C, and [0.05,0.2] for ε. The computer we used for these numerical experiments is PIII 450 PC and the operating system is windows 2000.

4.1 The *sinc* data

The function $\mathrm{sinc}(x)=|x|^{-1}\sin|x|$ is commonly used to illustrate SVR (Vapnik, 1995). Training and testing dataset are obtained by uniformly sampling 100 data points from the interval $[-10,10]$ respectively; the targets are corrupted by noise with normal distribution. We set the standard deviation σ of the additive noise at different levels and make a comparison of the inference results.

Table 1. Training results on *sinc* datasets at different noise level σ with a fixed β=0.3. ASE denotes the average squared error on the testing dataset.

σ	C	$\varepsilon\times10^2$	w	$-\ln P_{D\mid\theta}$	ASE
0	364.7	0. 0335	6.81	- 442.72	2.9×10^{-8}
0.1	6.052	5.3361	5.65	- 48.59	0.010983
0.2	3.698	6.8955	6.57	18.34	0.049409
0.3	1.926	6.3703	5.52	36.78	0.106605

We find that C and ε changed greatly according to the noise level. ARD parameter w is mainly determined by the input distribution. At the same time, the training machine becomes less confident on the hyperparameter settings.[3] ASE is approximately equal to noise variance σ^2.

Table 2. Training results on *sinc* dataset with noise standard deviation σ = 0.1 at different β. SV_M denotes the number of support vectors in $\Delta_{M^*} \cup \Delta_M$; AAE is the average absolute error on the test dataset.

β	C	$\varepsilon\times10^2$	w	$-\ln P_{D\mid\theta}$	SV_M	AAE	ASE$\times10^2$
.005	9.01	11.2	3.01	-31.5	7	.0830	1.071
0.01	5.57	8.58	3.86	-40.0	8	.0816	1.048
0.05	8.50	15.1	6.70	-46.1	11	.0819	1.101
0.1	6.63	16.6	5.09	-50.6	10	.0820	1.102
0.3	6.05	5.34	5.65	-48.6	18	.0834	1.098
0.5	5.98	3.89	5.32	-47.3	25	.0835	1.099
0.7	5.63	2.60	5.46	-48.3	22	.0839	1.113
1.0	5.78	3.89	4.97	-48.4	37	.0832	1.094

The choice of β has little influence on the training results except SV_M, the size of matrix Σ_M. The result verified our

[2] The determinant comes from $det(\Sigma)$ in Z_f multiplying $det(\Sigma^{-1}+C\Lambda)$ in $\int exp[-S(f)]df$, where I is the identity matrix.

[3] Constant term $n\cdot\ln 2$ in $-\ln P(D|\theta)$ (46) is excluded from the results in tables.

analysis that parameter β controls the sparseness in training, and the training results are insensitive to its value in the region where the matrix Σ_M has not become heavily shrunk.

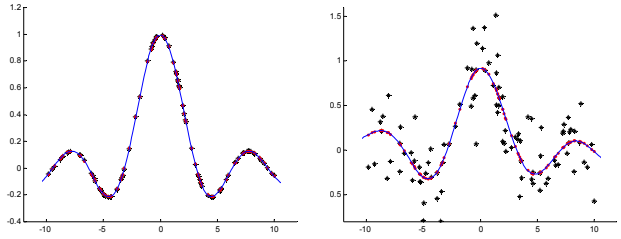

Figure 3. Graphs of part results in table 1. Left: clean training dataset, σ = 0; right: σ = 0.3. β is fixed at 0.3.

4.2 Robot Arm

The task in the robot arm problem is to learn the mapping from joint angles to position for an imaginary "robot arm". There are two real input variables, x_1 and x_2, representing joint angles, and two real target values, y_1 and y_2, representing the resulting arm position in rectangular coordinates. The actual relationship between inputs and targets is as follows:

$$\begin{aligned} y_1 &= 2.0\cos x_1 + 1.3\cos(x_1 + x_2) \\ y_2 &= 2.0\sin x_1 + 1.3\sin(x_1 + x_2) \end{aligned} \tag{47}$$

The dataset of robot arm problem contains 600 input-target pairs, which were randomly generated by picking x_1 uniformly from the ranges [-1.932, -0.453] and [+0.453, +1.932], and x_2 uniformly from the ranges [0.534, 3.142]. Targets are contaminated by independent Gaussian noise of standard deviation 0.05. The first 200 examples in the dataset are used as training set in all cases; the second 200 examples used as testing set 1 and the last 200 as testing set 2. [4]

Table 3. Results of training on the first 200 examples with a fixed β=0.3, and other hyperparameters are randomly initialized.

y	C	$\varepsilon_{\times 10^2}$	w_1	w_2	$-\ln P_{D\|\theta}$
y_1	28.326	0.2116	0.8079	0.2388	-399.70
y_2	29.423	0.1912	0.7764	0.2040	-471.24

Since the distribution of input points and the additive noise are same for two targets, it is reasonable that they possess almost same hyperparameter settings. UNIT TIME that the CPU needs to evaluate the evidence and all the gradients once is about 11.19 seconds, and TOTAL TIME for the whole training is usually about 627.39 seconds, which is mainly determined by the evaluation times required in the optimization package.

[4] The dataset of robot arm problem generated by Mackay is available at http://wol.ra.phy.cam.ac.uk/mackay/bigback/dat/.

Table 4. Test results with hyperparameter settings stated in table 3. ASN is the average squared additive noise, AAN is the average absolute additive noise. [5]

TESTING SET	$\text{ASN}_{\times 10^3}$	$\text{ASE}_{\times 10^3}$	AAN	AAE
SET 1 ON y_1 ONLY	2.255	2.5092	0.0383	0.03929
SET 1 ON y_2 ONLY	2.787	3.2021	0.0427	0.04602
SET 2 ON y_1 ONLY	2.207	2.7751	0.0384	0.04334
SET 2 ON y_2 ONLY	2.173	2.2717	0.0383	0.03859
SET 1 ON POSITION	5.042	5.7113	0.0641	0.06711
SET 2 ON POSITION	4.380	5.0467	0.0597	0.06401

We compare the test error with other implementations, such as in neural networks, Gaussian approximation by Mackay (1992) and Monte Carlo by Neal (1996), and Gaussian processes for regression by Williams & Rasmussen (1996). The expected test error for guesses based on knowledge of the true distribution should be 0.005.

Table 5. Average squared error (ASE) of testing on the robot arm positions comparing with other implementations.

IMPLEMENTATION METHOD	ASE
GAUSSIAN APPROXIMATION OF MACKAY	
SOLUTION WITH HIGHEST EVIDENCE	0.00573
SOLUTION WITH LOWEST TEST ERROR	0.00557
HYBRID MONTE CARLO WITH 150 SUPER-TRANSITIONS	
LAST 100 POINTS FROM INDIVIDUAL RUNS	0.00554
LAST 100 POINTS FROM THREE RUNS	0.00557
GAUSSIAN PROCESSES OF WILLIAMS	0.00563
TRAINING METHOD PROPOSED IN THIS PAPER	0.005379

4.3 Boston Housing

The data concerns the median price in 1970 of owner-occupied houses in 506 census tracts within the Boston metropolitan area. Thirteen attributes pertaining to each census tract are available for use in prediction. [6] Following the method used by Tipping (2000) and Saunders, Gammerman & Vovk (1998) [7], the dataset is partitioned into 481/25 training/testing splits randomly. This partitioning is carried out 100 times on the data.

[5] Since the estimate of robot arm position involves two dimensions in rectangular coordinates, y_1 and y_2, the test errors on only one dimension are shown first, and then we calculate the estimate error on the position.

[6] The original data is in StatLib, available at URL http://lib.stat.cmu.edu/datasets/boston.

[7] In this paper, the validation set, 80 cases in training set, is used to determine the kernel parameters.

Table 6. The average and standard deviation of ASE, NMSE, AAE and UNIT TIME, TOTAL TIME in seconds over the 100 trials. β is fixed at 0.3, and other hyperparameters are randomly initialized.

	ASE	NMSE	AAE	UNIT TIME	TOTAL TIME
MEAN	8.03	0.104	1.981	111.035	2775.786
STD	4.11	0.059	0.368	17.906	611.043

Table 7. A comparison with Ridge Regression (Saunders et al., 1998) and the Relevance Vector Machine (Tipping, 2000).

METHOD	KERNEL TYPE	ASE
RIDGE REGRESSION	POLYNOMIAL	10.44
RIDGE REGRESSION	SPLINES	8.51
RIDGE REGRESSION	ANOVA SPLINES	7.69
RELEVANCE VECTOR	LINEAR SPLINES	7.46
PROPOSED IN THE PAPER	GAUSSIAN	8.03

The difference in performance may be due to the random selection of the training/testing sets and the different kernel we used.

5. Conclusion

SILF inherits all the virtues of popular non-quadratic loss functions. The properties of SILF are entirely controlled by two parameters β and ε. β stands for a tradeoff between Huber's loss function and Vapnik's ILF; ε represents the biasness in noise distribution. The introduction of SILF into likelihood function for regression problem opens a way to soundly integrate Bayesian inference with Support Vector methods in probabilistic framework. Global minimum is guaranteed when we determine the *MAP* point using Support Vector methods. Then we resort to Laplacian approximation, and model adaptation can be done by gradient-based optimization methods on the criterion of the evidence. Furthermore, sparse matrix in the evidence calculation accelerates training speed and helps us to tackle relatively huge dataset. However, when the number of hyperparameters increases, training process becomes more likely to be stuck at different local minimum. In general we can train several times starting from different initial states, and choose the one with highest evidence as optimal choice while discarding other candidates. We can also use these candidates to build up a mixture of experts to improve the performance.

Acknowledgements

Wei Chu gratefully acknowledges the financial support provided by the National University of Singapore through Research Scholarship. The reviewers' thoughtful comments are gratefully appreciated.

References

Wei Chu (2001). *Support Vector methods and Bayesian inference in Gaussian processes for regression.* Technical Report, Control Division, Department of Mechanical Engineering, National University of Singapore, available at http://guppy.mpe.nus.edu.sg/~mpessk/svm/report.pdf

Federico Girosi (1991). *Models of noise and robust estimates. A.I. Memo No. 1287.*

J. T. Kwok (1999). *Integrating the evidence framework and the Support Vector machine.* In *Proceedings of the European Symposium on Artificial Neural Networks*, pages 177-182, Bruges, Belgium

P. J. Huber (1972). *Robust statistics: a Review Ann. Statist. 43:1041.*

Mackay D. J. C. (1992). *A practical Bayesian framework for back propagation networks. Neural Computation, 4(3)*, 448-472.

Mackay D. J. C. (1993). *Bayesian methods for backpropagation Networks.* In J. L. van Hemmen, E. Domany, and K. Schulten, editors, *Models of Neural Networks II.* Springer.

Radford M. Neal (1996). *Bayesian learning for neural networks.* Lecture Notes in Statistics, Springer.

John C. Platt (1999). *Probabilistic outputs for Support Vector Machines and comparisons to regularized likelihood methods.* In A.J. Smola, P.Bartlett, B. Schoelkopf, and C. Schuurmans, editors, *Advanced in Large Margin Classifiers*, MIT Press.

Massimiliano Pontil, Sayan Mukherjee & Federico Girosi (1998). *On the noise model of Support Vector machine regression. A.I. Memo No. 1651*, Artificial Intelligence Laboratory, Massachusetts Institute of Technology.

P. Sollich (2001). *Bayesian methods for support vector machines: evidence and predictive class probabilities. Machine Learning*, to appear.

C. Saunders, A. Gammerman, V. Vovk (1998). *Ridge regression learning algorithm in dual variables.* In *Proceedings of the 15th International Conference on Machine Learning*, pages 515-521

Michael E. Tipping (2000). *The relevance vector machine.* In S. A. Solla, T. K. Leen, and K.-R. Müller (Eds.), *Advances in Neural Information Processing Systems 12*, pp. 652–658. Cambridge, MIT Press.

Vapnik Vladimir N. (1995). *The nature of statistical learning theory.* Springer.

C. K. I. Williams and Carl Edward Rasmussen (1996). *Gaussian processes for regression.* In D. S. Touretzky, M. C. Mozer, and M. E. Hasselmo, editors, *Advanced in Neural Information Processing Systems 8*. MIT Press.