A NEW BAYESIAN DESIGN METHOD FOR SUPPORT VECTOR CLASSIFICATION

Wei Chu, S. Sathiya Keerthi, Chong Jin Ong

Control Division, Department of Mechanical Engineering, National University of Singapore
10 Kent Ridge Crescent, Singapore, 119260
engp9354@nus.edu.sg, mpessk@nus.edu.sg, mpeongcj@nus.edu.sg

ABSTRACT

In this paper, we apply popular Bayesian techniques on support vector classifier. We propose a novel differentiable loss function called trigonometric loss function with the desirable characteristic of natural normalization in the likelihood function, and then follow standard Gaussian processes techniques to set up a Bayesian framework. In this framework, Bayesian inference is used to implement model adaptation, while keeping the merits of support vector classifier, such as sparseness and convex programming. Moreover, we put forward class probability in making predictions. Experimental results on benchmark data sets indicate the usefulness of this approach.

1. INTRODUCTION

As a computationally powerful class of supervised learning networks, classical support vector classifier (SVC) exploits the idea of mapping the input data into a high dimensional (often infinite) Hilbert space defined by a reproducing kernel (RKHS), where a linear classification is performed. The discriminant function is constructed by solving a regularized functional via convex quadratic programming. The choice of the regularization parameter and the other kernel parameters in the SVC model crucially affect the generalization performance. Model selection is usually based on the criterion of some simple and pertinent performance measures, such as cross validation or various generalization bounds derived from statistical learning theory. Typically, Bayesian methods are regarded as suitable tools to determine the values of these parameters. Moreover, Bayesian methods can also provide probabilistic class prediction that is more desirable than just deterministic classification.

There is some literature on Bayesian interpretations of classical SVC. Kwok [3] built up MacKay's evidence framework [4] using a weight-space interpretation. The unnormalized evidence may cause inaccuracy in Bayesian inference. Sollich [6] pointed out that the normalization issue in Bayesian framework for classical SVC is critical and proposed an intricate Bayesian treatment with normalized evidence and error bar, where the evidence normalization depends on an unknown input distribution that limits its usefulness in practice.

In this paper, we introduce a novel loss function for SVC, called the trigonometric loss function, with the purpose of integrating Bayesian inference with SVC smoothly while preserving their individual merits. The trigonometric loss function is smooth and naturally normalized in likelihood evaluation. Further, it possesses the desirable property of sparseness in sample selection. We follow standard Gaussian processes for classification [7] to set up a Bayesian framework. Maximum a posteriori (MAP) estimate of the latent functions results in a convex programming problem. The popular sequential minimal optimization algorithms could be easily adapted to find the solution. Optimal parameters can then be inferred by Bayesian techniques with the benefit of sparseness, and probabilistic class prediction can also be provided for test patterns.

2. TRIGONOMETRIC LOSS FUNCTION

In the probabilistic approach for binary classification, logistic function is widely used as an approximation for the discontinuous heaviside step function in likelihood evaluation [7]. The logistic function is defined as

$$\mathcal{P}(y_x|f_x) = \frac{1}{1+\exp(-y_x \cdot f_x)} \tag{1}$$

where the input vector $x \in \Re^d$, the class label $y_x \in \{+1, -1\}$ and f_x denotes the latent function (discriminant function) at x. $-\ln \mathcal{P}(y_x|f_x)$ is usually referred to as loss function. The loss function associated with the shifted heaviside step function in classical SVC is also called hard margin loss function, which is defined as

$$\ell_h(y_x \cdot f_x) = \begin{cases} 0 & if \quad y_x \cdot f_x \geq +1; \\ +\infty & otherwise. \end{cases} \tag{2}$$

The hard margin loss function is suitable for noise-free data sets. For other general cases, a soft margin loss function is

Wei Chu gratefully acknowledges the financial support provided by the National University of Singapore through Research Scholarship.

popularly used in classical SVC, which is defined as

$$\ell_\rho(y_x \cdot f_x) = \begin{cases} 0 & if \quad y_x \cdot f_x \geq +1; \\ (1 - y_x \cdot f_x)^\rho & otherwise, \end{cases} \tag{3}$$

where ρ is a positive integer. The corresponding likelihood function in probabilistic framework could be written as

$$\mathcal{P}(y_x|f_x) = \frac{1}{\nu(f_x)} \cdot \exp(-\ell_\rho(y_x \cdot f_x)),$$

where $y_x \in \{-1, +1\}$ and the normalizer should be $\nu(f_x) = \exp(-\ell_\rho(+f_x)) + \exp(-\ell_\rho(-f_x))$. Notice that the normalizer $\nu(f_x)$ is dependent on the latent function f_x. This flaw precludes the solution of SVC from being directly used as the MAP estimate on function values in Bayesian inference [6]. The loss functions in SVC are special in that they give identical zero penalty to training samples that have satisfied the constraint $y_x \cdot f_x > +1$. These training samples are not involved in the Bayesian inference computations. The simplification of computational burden is usually referred to as the sparseness property. Logistic function (1) does not enjoy this property since it contributes a positive penalty to all the training samples. On the other hand, logistic function is attractive because it is naturally normalized in likelihood evaluation, i.e., the normalizer is a constant, a property that allows Bayesian techniques to be used smoothly.

Based on these observations, we generalize the desirable characteristics in these loss functions for classification: it should be naturally normalized in likelihood evaluation; it should possess a flat zero region that results in sparseness property; it should be smooth and its first order derivative should be explicit and simple. Adhering to these requirements, we propose a novel loss function for binary classification, known as trigonometric loss function. The trigonometric loss function is defined as

$$\ell_t(\delta) = \begin{cases} +\infty & if \quad \delta \in (-\infty, -1]; \\ 2 \ln \sec(\frac{\pi}{4}(1-\delta)) & if \quad \delta \in (-1, +1); \\ 0 & if \quad \delta \in [+1, +\infty), \end{cases} \tag{4}$$

where $\delta = y_x \cdot f_x$. The trigonometric likelihood function is therefore written as

$$\mathcal{P}_t(y_x|f_x) = \begin{cases} 0 & if \quad \delta \in (-\infty, -1]; \\ \cos^2(\frac{\pi}{4}(1-\delta)) & if \quad \delta \in (-1, +1); \\ 1 & if \quad \delta \in [+1, +\infty). \end{cases} \tag{5}$$

The derivatives of the loss function are needed in the implementation of Bayesian methods. The first order derivative of (4) with respect to f_x can be derived as

$$\frac{\partial \ell_t(\delta)}{\partial f_x} = \begin{cases} -y_x \frac{\pi}{2} \tan(\frac{\pi}{4}(1-\delta)) & if \quad \delta \in (-1, +1); \\ 0 & if \quad \delta \in [+1, +\infty), \end{cases}$$

and the second order derivative is

$$\frac{\partial^2 \ell_t(\delta)}{\partial f_x^2} = \begin{cases} \frac{\pi^2}{8} \sec^2(\frac{\pi}{4}(1-\delta)) & if \quad \delta \in (-1, +1); \\ 0 & if \quad \delta \in [+1, +\infty). \end{cases} \tag{6}$$

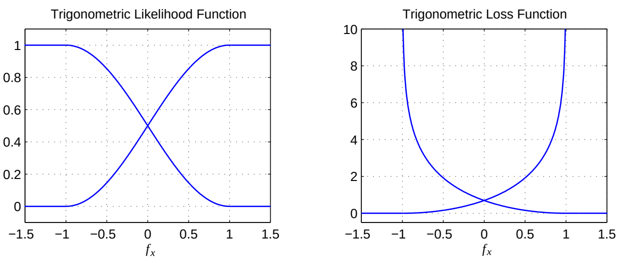

Figure 1: The graphs of trigonometric likelihood function and its loss function. The horizontal axis indicates the latent function f_x of the input vector x.

From the definition (5) and Figure 1, it is easy to see that the normalizer $\nu(f_x)$ is a constant for any f_x. From the definition (4) and Figure 1, we find that the trigonometric loss function possesses a flat zero region that is same as the loss functions in classical SVC, but it requires that $y_x \cdot f_x > -1$ should always hold.

3. BAYESIAN INFERENCE

Substituting the trigonometric loss function for the loss function in classical SVC, the trigonometric SVC (TSVC) minimizes the following regularized functional in a RKHS

$$\min_{\boldsymbol{f} \in \text{RKHS}} \mathcal{R}(\boldsymbol{f}) = \sum_{i=1}^{n} \ell_t(y_{x_i} \cdot f_{x_i}) + \lambda \|\boldsymbol{f}\|^2_{\text{RKHS}}, \tag{7}$$

where the regularization parameter λ is positive and $\|\boldsymbol{f}\|^2_{\text{RKHS}}$ is a norm in the RKHS. The function $\boldsymbol{f}$ could be also explained as a family of random variables in a Gaussian process due to the duality between RKHS and stochastic processes. Recently, Gaussian processes have provided a promising non-parametric Bayesian approach to classification problems [7]. The important advantage of Gaussian process models over other non-Bayesian models is the explicit probabilistic formulation. This not only builds the ability to infer model parameters in Bayesian framework but also provides probabilistic class prediction.

3.1. Bayesian Framework

We follow the standard Gaussian process classifier to describe a Bayesian framework, in which we impose a Gaussian process prior distribution on the latent functions and employ the trigonometric loss function in likelihood evaluation. This classifier, TSVC in the Bayesian framework, is referred to as Bayesian TSVC (BTSVC). The functions $\boldsymbol{f}$ are usually assumed as the realizations of random variables indexed by the input vector x_i in a stationary zero-mean

Gaussian stochastic process. The Gaussian process can then be specified by giving covariance matrix for any finite set of zero-mean random variables $\{f(x_i)|i = 1, 2, \ldots, n\}$. The covariance between the outputs corresponding to the inputs x_i and x_j could be defined as

$$Cov(x_i, x_j) = \kappa_0 \exp\left(-\frac{1}{2}\kappa\|x_i - x_j\|^2\right) + \kappa_b, \quad (8)$$

where $\kappa_0 > 0$ and $\kappa > 0$.[1] With ARD parameters, Gaussian covariance function could be enhanced as

$$Cov(x_i, x_j) = \kappa_0 \exp\left(-\frac{1}{2}\sum_{\iota=1}^{d}\kappa_\iota(x_i^\iota - x_j^\iota)^2\right) + \kappa_b, \quad (9)$$

where x^ι denotes the ι-th entry of the input vector x, and κ_ι is the ARD parameter and it determines the relevance of the ι-th input dimension to the target.

We collect the parameters in the prior distribution, as θ, the hyperparameter vector. Thus, for a given hyperparameter vector θ, the prior probability of the random variables $\{f(x_i)\}$ is a multivariate Gaussian, which can be written as

$$\mathcal{P}(f\,|\theta) = \frac{1}{Z_f}\exp\big(-\frac{1}{2}f^T\Sigma^{-1}f\big), \quad (10)$$

where $f = [f(x_1), f(x_2), \ldots, f(x_n)]^T$, Σ is the $n \times n$ covariance matrix whose ij-th element is $Cov(x_i, x_j)$, and $Z_f = (2\pi)^{\frac{n}{2}}|\Sigma|^{\frac{1}{2}}$.

The likelihood with the trigonometric likelihood function (5) can be written as

$$\mathcal{P}(D|f, \theta) = \prod_{i=1}^{n}\mathcal{P}_t(y_{x_i}|f(x_i)). \quad (11)$$

Based on Bayes' theorem, the posterior probability of f can then be written as

$$\mathcal{P}(f\,|D, \theta) = \frac{1}{Z_S}\exp\left(-S(f)\right), \quad (12)$$

where $S(f) = \frac{1}{2}f^T\Sigma^{-1}f + \sum_{i=1}^{n}\ell_t(y_{x_i}\cdot f(x_i))$, $\ell_t(\cdot)$ is defined as in (4), and $Z_S = \int\exp(-S(f))\,df$. Since $\mathcal{P}(f|D,\theta) \propto \exp(-S(f))$, the MAP estimate on the values of f is therefore the minimizer of the following optimization problem

$$\min_f \mathcal{S}(\boldsymbol{f}) = \frac{1}{2}f^T\Sigma^{-1}f + \sum_{i=1}^{n}\ell_t(y_{x_i}\cdot f(x_i)). \quad (13)$$

In the regularized functional, κ_0 in covariance matrix plays the role as the regularization parameter.[2]

[1] Note that the exponential term in (8) is exactly the Gaussian kernel in classical SVC. Other kernel functions in classical SVC could also be used in the covariance function, such as the polynomial and spline kernel function.

[2] Note that there is an equivalence between the regularized functional (13) and that of TSVC in the RKHS (7).

3.2. Convex Programming

In this subsection, we formulate the optimization problem (13) as a convex programming problem. As usual, slack variables ξ_i are introduced: $\xi_i \geq 1 - y_{x_i}\cdot f(x_i)$, $\forall i$. The optimization problem (13) can then be restated as the following equivalent optimization problem, which we refer to as the *primal* problem:

$$\min_{f,\xi}\frac{1}{2}f^T\Sigma^{-1}f + 2\sum_{i=1}^{n}\ln\sec\left(\frac{\pi}{4}\xi_i\right) \quad (14)$$

subject to $y_{x_i}\cdot f(x_i) \geq 1 - \xi_i$ and $0 \leq \xi_i < 2$, $\forall i$. Standard Lagrangian techniques are used to derive the *dual* problem. The strict inequality $\xi_i < 2$ is assumed to hold and omitted. As we will see below, this condition will be implicitly taken into account in the solution. Let $\alpha_i \geq 0$ and $\gamma_i \geq 0$ be the corresponding Lagrange multipliers for other inequalities in the *primal* problem (14). The KKT conditions for the *primal* problem (14) require

$$f(x_i) = \sum_{j=1}^{n}y_{x_j}\alpha_j Cov(x_i, x_j) \qquad \forall i; \quad (15)$$

$$\xi_i = \frac{4}{\pi}\arctan\left(\frac{2}{\pi}(\alpha_i + \gamma_i)\right) \qquad \forall i. \quad (16)$$

Note that with ξ_i defined as in (16), the condition $\xi_i < 2$ is automatically taken into account. The *dual* problem can be finally written as

$$\min_\alpha \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n}(y_{x_i}\alpha_i)(y_{x_j}\alpha_j)Cov(x_i, x_j) - \sum_{i=1}^{n}\alpha_i + \sum_{i=1}^{n}\left[\frac{4}{\pi}\alpha_i\arctan\left(\frac{2\alpha_i}{\pi}\right) - \ln\left(1 + \left(\frac{2\alpha_i}{\pi}\right)^2\right)\right] \quad (17)$$

subject to $\alpha_i \geq 0$, $\forall i$.

The popular sequential minimal optimization algorithm for classical SVC [2] can be easily adapted to solve the optimization problem. Other methods for solving convex programming problems, such as dual subgradient schemes, can also be used for the solution. At the optimal solution to (17), the MAP estimate on the values of the random variables f can be written in column vector form

$$f_{\mathrm{MP}} = \Sigma\cdot w \quad (18)$$

where $w = [y_{x_1}\alpha_1, y_{x_2}\alpha_2, \ldots, y_{x_n}\alpha_n]^T$. The training samples (x_i, y_{x_i}) associated with non-zero Lagrange multiplier α_i are called *support vectors* (SVs). The other samples associated with zero α_i do not involve in the solution representation and the following Bayesian computation. This property is usually referred to as sparseness, and it reduces the computational cost significantly.

3.3. Hyperparameter Inference

The optimal values of hyperparameters θ can be inferred by maximizing the posterior probability $\mathcal{P}(\theta|D)$, using $\mathcal{P}(\theta|D) = \mathcal{P}(D|\theta)\mathcal{P}(\theta)/\mathcal{P}(D)$. A prior distribution on the hyperparameters $\mathcal{P}(\theta)$ 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 $\mathcal{P}(\theta)$, i.e., $\mathcal{P}(\theta)$ is greatly insensitive to the values of θ. Therefore, $\mathcal{P}(D|\theta)$, known as the evidence of θ, can be used to assign a preference to alternative values of the hyperparameters θ [4]. The evidence could 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.

We can get the evidence by an integral over all f: $\mathcal{P}(D|\theta) = \int \mathcal{P}(D|\theta, f)\mathcal{P}(f|\theta)\, df$. Using the definitions in (10) and (11), the evidence can also be written as

$$\mathcal{P}(D|\theta) = \frac{1}{Z_f} \int \exp(-S(f))\, df. \tag{19}$$

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

$$S(f) \approx S(f_{\mathrm{MP}}) + \frac{1}{2}(f - f_{\mathrm{MP}})^T \left.\frac{\partial^2 S(f)}{\partial f \partial f^T}\right|_{f=f_{\mathrm{MP}}} (f - f_{\mathrm{MP}}), \tag{20}$$

where $\frac{\partial^2 S(f)}{\partial f \partial f^T} = \Sigma^{-1} + \Lambda$, and Λ is a diagonal matrix coming from the second order derivative of trigonometric loss function (6). Introducing (20) into (19) yields $\mathcal{P}(D|\theta) = \exp(-S(f_{\mathrm{MP}})) \cdot |I + \Sigma \cdot \Lambda|^{-\frac{1}{2}}$, where I is the identity matrix. Notice that only a sub-matrix of Σ plays a role in the determinant $|I + \Sigma \cdot \Lambda|$ due to the sparseness of the diagonal matrix Λ in which only the entries associated with SVs are non-zero. We denote their sub-matrices as Σ_{M} and Λ_{M} respectively by keeping their non-zero entries. From (18), we can write the MAP estimate of f in vector form as $f_{\mathrm{MP}} = \Sigma \cdot w$, where $w = [y_{x_1}\alpha_1, y_{x_2}\alpha_2, \ldots, y_{x_n}\alpha_n]^T$. We denote w_{M} as the sub-vector of w by keeping entries associated with SVs. Because of these sparseness properties, the negative log of the evidence can then be simplified as

$$-\ln \mathcal{P}(D|\theta) = \frac{1}{2} w_{\mathrm{M}}^T \cdot \Sigma_{\mathrm{M}} \cdot w_{\mathrm{M}} + 2 \sum_{m \in \mathrm{SVs}} \ln \sec\left(\frac{\pi}{4}\xi_m\right) + \frac{1}{2} \ln |I + \Sigma_{\mathrm{M}} \cdot \Lambda_{\mathrm{M}}|, \tag{21}$$

where $\xi_m = 1 - y_{x_m} \cdot f_{\mathrm{MP}}(x_m)$, $\forall m$. The evidence evaluation is a convenient yardstick for model selection. The minimizer of $-\ln \mathcal{P}(D|\theta)$ (21) could be inferred by some gradient-based optimization methods, since it is easy to derive the gradient with respect to θ. In standard Gaussian processes for classification [7], the inversion of the full matrix Σ has to be computed in iterative mode. This is a heavy burden for large-scale learning tasks. In our approach, only the inversion of the sub-matrix Σ_{M}, corresponding to the SVs, is required in the gradient evaluation. This sparseness in gradient evaluation makes it possible for our approach to tackle reasonably large data sets with thousands of samples, as the SVs are usually a small subset of the training samples.

4. PROBABILISTIC CLASS PREDICTION

In this section, we present the probabilistic class prediction on test patterns. This ability to provide the class probability is one of the important advantages of our probabilistic approach over the usual deterministic approach.

Let us take a test case x for which the class label y_x is unknown. The random variable $f(x)$ index by x along with the n zero-mean random variables f have a joint $n+1$ multivariate Gaussian distribution, and then the conditional distribution of $f(x)$ given f is also a Gaussian

$$\mathcal{P}(f(x)|f, D, \theta) \propto \exp\left(-\frac{1}{2}\frac{(f(x) - f^T\Sigma^{-1}\boldsymbol{k})}{Cov(x,x) - \boldsymbol{k}^T\Sigma^{-1}\boldsymbol{k}}\right) \tag{22}$$

where $\boldsymbol{k} = [\,Cov(x, x_1), \ldots, Cov(x, x_n)\,]^T$. To erase the uncertainty in f, we compute $\mathcal{P}(f(x)|D,\theta)$ by an integral over f-space. The integral could be approximately evaluated as (see [1] for more details)

$$\mathcal{P}(f(x)|D,\theta) \sim N(\mu, \sigma_t^2) = \frac{1}{\sqrt{2\pi}\sigma_t}\exp\left(-\frac{(f(x)-\mu)^2}{2\sigma_t^2}\right) \tag{23}$$

with the variance $\sigma_t^2 = Cov(x,x) - \boldsymbol{k}_{\mathrm{M}}^T(\Lambda_{\mathrm{M}}^{-1} + \Sigma_{\mathrm{M}})^{-1}\boldsymbol{k}_{\mathrm{M}}$ and the mean $\mu = w_{\mathrm{M}}^T\boldsymbol{k}_{\mathrm{M}}$, where $\boldsymbol{k}_{\mathrm{M}}$ is the sub-vector of $\boldsymbol{k}$ by keeping the entries associated with SVs.

Now we make probabilistic class prediction. Given the hyperparameters θ, the probability of the binary class label y_x for the testing case x can be evaluated as:

$$\mathcal{P}(y_x|D,\theta) = \int \mathcal{P}(y_x|f(x), D, \theta)\mathcal{P}(f(x)|D,\theta)\, df(x)\,,$$

where $\mathcal{P}(y_x|f(x), D, \theta)$ is evaluated by trigonometric likelihood function (5) and $\mathcal{P}(f(x)|D,\theta)$ is given by (23). The one dimensional integral is easy to be computed as:

$$\mathcal{P}(y_x|D,\theta) = \frac{1}{2}\mathrm{erfc}\left(\frac{1 - y_x\mu}{\sqrt{2}\sigma_t}\right) + \int_{-1}^{+1} \cos^2\left(\frac{\pi}{4}(1 - y_x f(x))\right) N(\mu, \sigma_t^2)\, df(x)\,, \tag{24}$$

where $\mathrm{erfc}(\nu) = \frac{2}{\sqrt{\pi}}\int_\nu^{+\infty}\exp(-z^2)\,dz$.

Table 1: Training results of BTSVC with Gaussian covariance function (8) on benchmark data sets. Splice* denotes the reduced Splice data set. TIME denotes the average CPU time in seconds consumed by Bayesian inference over the first five folds; RATE denotes the test error rate in percent averaged over all folds of that data set; SVC is the RATE of classical SVC with Gaussian kernel reported in [5].

Data Set	TIME	RATE	SVC
Banana	8.65±2.78	**10.39±0.50**	11.53±0.66
Breast	3.24±0.64	**25.70±4.46**	26.04±4.74
Diabetis	22.43±5.61	**23.13±1.75**	23.53±1.73
Flare	85.15±12.26	34.26±1.75	**32.43±1.82**
German	106.35±26.96	**23.37±2.28**	23.61±2.07
Heart	1.23±0.64	16.33±2.78	**15.95±3.26**
Image	71.85±28.45	3.50±0.62	**2.96±0.60**
Ringnorm	4.50±1.10	1.99±0.26	**1.66±0.12**
Splice	308.22±87.41	12.36±0.72	**10.88±0.66**
Splice*	42.00±14.64	**5.87±0.58**	—
Thyroid	0.21±0.04	**3.95±2.07**	4.80±2.19
Titanic	1.29±0.47	22.51±1.01	**22.42±1.02**
Twonorm	1.19±0.35	**2.90±0.27**	2.96±0.23
Waveform	4.52±2.76	9.94±0.42	**9.88±0.43**

5. NUMERICAL EXPERIMENTS

In numerical experiments, the initial states of the hyperparameters are chosen as $\kappa_b = 100.0$ and $\kappa = 1/d$, where d is the input dimension. The initial value of κ_0 is chosen from $\{0.1, 1, 10, 100\}$, usually it is 10. The computer we used for these numerical experiments is PIII 866 PC with 384MB RAM and the operating system is Windows 2000.[3] We carried out Bayesian inference with Gaussian kernel on the benchmark data sets used by Rätsch et al. [5].[4] We report the training results of BTSVC on these data sets in Table 1. The optimal hyperparameters used throughout the training on all folds of that data set is determined by the average results of Bayesian inference on the first five folds. We choose optimal hyperparameters in this way for a fair comparison with the results of classical SVC reported in [5]. We find that the generalization capability of our Bayesian approach is very competitive. In next series of experiments, we choose the Splice data, which has the highest dimension in these benchmarking data, to carry out ARD feature selection. Bayesian inference with ARD Gaussian kernel (9) yields more preferable evidence and reduces test error rate to 5.29%. From the optimal ARD parameters, we find that only the $28^{\rm th} - 34^{\rm th}$ input dimensions are significantly relevant in the whole 60 dimensions. Thus, we create reduced Splice data sets by keeping the 7 relevant dimensions only. On the reduced data sets, Gaussian covariance kernel can still yields similar performance. It is one of the advantages of Bayesian design over the deterministic approach that large number of hyperparameters can be tuned systematically.

[3]The program we used in the numerical experiments is available at www.guppy.mpe.nus.edu.sg/ mpessk/svm/bisvm.zip.

[4]These 100-fold benchmark data sets (only 20 folds available for Image and Splice) and related experimental results can be accessed at http://www.first.gmd.de/∼raetsch/data/benchmarks.htm.

6. CONCLUSION

We proposed a Bayesian support vector classifier by introducing trigonometric likelihood function. In the probabilistic framework of stationary Gaussian processes, various computational procedures are provided for the MAP estimate and the evidence of the hyperparameters. Model adaptation and ARD feature selection could be implemented intrinsically in hyperparameter inference. Furthermore, the sparseness reduces the computational cost significantly. Another benefit is the availability of class probabilities in making predictions. The results in numerical experiments verify that the generalization capability is competitive. We will further study the influence from outliers in future work.

7. REFERENCES

[1] W. Chu, S. S. Keerthi, and C. J. Ong, Bayesian trigonometric support vector classifier. *Technical Report, CD-01-16* Dept. of Mechanical Engineering, National University of Singapore, 2001. http://guppy.mpe.nus.edu.sg/∼mpessk/btsvc/btsvc.pdf

[2] S. S. Keerthi, S. K. Shevade, C. Bhattacharyya, and K. R. K. Murthy, Improvements to Platt's SMO algorithm for SVM classifier design, *Neural Computation*, Vol. 13, 637-649, 2001.

[3] J. T. Kwok, The evidence framework applied to support vector machines. *IEEE Transactions on Neural Networks* , 11(5):1162-1173, 2000.

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

[5] G. Rätsch, T. Onoda, and K.-R. Müller, Soft margins for AdaBoost. *Machine Learning*. 42(3), 287-320, 2001.

[6] P. Sollich, Bayesian methods for Support Vector Machines: Evidence and predictive class probabilities. *Machine learning*, 46:21-52, 2002.

[7] C. K. I. Williams and D. Barber, Bayesian classification with Gaussian processes. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 20(12), 1342-1351, 1998.