

Chapter 1

Combining Information with a Bayesian Multi-class Multi-kernel Pattern Recognition Machine

Theodoros Damoulas and Mark A. Girolami

Inference Group,

*Department of Computing Science, FIMS, University of Glasgow,
18 Lilybank Gardens, SWA Building, G12 8QQ, Glasgow, Scotland, UK,
{theo, girolami}@dcs.gla.ac.uk*

In this contribution we offer a multi-class multi-kernel machine based on the multinomial probit likelihood which is able to informatively combine diverse sources of information and multiple feature spaces. The proposed methodology follows a well-founded hierarchical Bayesian paradigm that models uncertainty in the parameters via a hierarchy of prior and hyper-prior distributions that are well described and justified. We offer the full Markov chain Monte Carlo (MCMC) solution via a Metropolis-Hastings (MH) within Gibbs sampling approach and a Variational Bayes (VB) approximation to the solution which enables efficient CPU times and reduced computational complexity. We also provide and examine different combination strategies that can be employed with the model, including a weighted summation, a binary switch and a weighted product rule. The proposed approach provides the current *state-of-the-art* in a variety of domains such as protein fold prediction, remote homology detection and handwritten numerals classification, and matches or outperforms previous ensemble learning methods that are based on the popular support vector machines (SVM) methodology. Finally, we examine the efficiency of the VB approximation against the full MCMC solution and the insight our method offers on these problems by inferring the significance of various sources of information and string kernels.

Contents

1.1 Introduction	2
1.2 Intuition and Motivation	5
1.3 Multinomial Probit Kernel Combination	5
1.3.1 Markov chain Monte Carlo solution	9

1.3.2	Variational Bayes approximation	11
1.4	Experiments and Results	14
1.4.1	Proof of concept on an artificial data-set	15
1.4.2	UCI and handwritten numerals data-sets	18
1.4.3	Protein fold recognition and remote homology detection	21
1.5	Discussion and Future Directions	24
	References	26

1.1. Introduction

Problem area

In a large number of pattern recognition and machine learning problems we encounter the situation where different sources of information and different representations are available for the specific object that we are trying to classify. For example consider different representations of an image (i.e. pixel grey-scale values, gabor coefficients, fourier coefficients), different attributes of a signal or even more heterogeneous sources such as an image of a physical object, its response to specific radiation and the concentration of iron in its constitution. In these cases the underlying problem is how to combine all the available sources, in an informative way, in order to achieve a superior classification performance for the task in hand while at the same time dealing with the “curse of dimensionality”^{5,7,30} inherited in the domain.

When multiple feature spaces/sources S are available for a multinomial classification task there are broadly three distinct approaches available:

a) Concatenate all the features together into a large dimensional space and employ a multinomial classifier^a. *b)* Employ a multinomial classifier^a on each feature space and then combine. *c)* Combine first the feature spaces and then employ a multinomial classifier^a.

First of all it is easy to understand the computational drawbacks if we employ binary classifiers instead of multinomials in all cases. Next, although concatenating might work fine in certain problems, the possible excessive dimensionality of the final augmented space will be forbidding for most problems and classifiers. At the same time, scaling issues and inability to control the contribution of each individual space is another problem, especially when suspect or degraded features might be included.

^aOr combine binary classifiers

Past methods

The typical preferred approach so far has been to combine multinomial or even binary classifiers, namely *ensemble learning* methods.¹³ The idea behind that approach is to train one classifier in every feature space and then combine their class predictive distributions. Different ways of combining the output of the classifiers have been studied^{21,22,41} and also meta-learning an overall classifier on these distributions⁴³ has been proposed.

The *drawbacks* of that approach lies on the theoretical justification, on the processing loads incurred as multiple training has to be performed and on the fact that the individual classifiers operate independently on the data. Their performance has been shown to significantly improve over the best individual classifier in many cases but the extra load of training multiple classifiers may possibly restrain their application when resources are limited. The typical combination rules for classifiers are *ad hoc* methods⁶ that are based on the notions of the *linear* or *independent* opinion pools.

Recently, *kernel combination* methodologies that follow the third approach, are being proposed that seem suitable for such problems. Previous work on kernel combination includes²³ where semidefinite programming is used to learn the composite kernel matrix,³⁴ where a hyper-kernel space is defined on the space of kernels in order to learn the composite kernel within a specific parametric family,²⁵ where Gaussian kernels under a Support Vector Machine (SVM) framework are combined into an expanded composite kernel,³⁵ in which the composite kernel is a summation of base kernels, without however learning the combinatorial weights or the kernel parameters and work by¹⁵ where hierarchic Bayesian models are employed to infer the combinatorial weights and perform kernel learning. Furthermore, in recent work by²⁷ a nonstationary kernel combination approach was presented which allows for variation on the relative weights of the base kernels among the input examples. In this case the kernel combination weights are a function of the input.

The *drawbacks* of the above methods are their foundation on binary classifiers by nature (mostly SVMs) that leads to bad scaling for multi-class problems (We need for example 4,350 classifiers for a 10-fold cross validation on a 30 class problem employing an all-versus-all method). Also, the majority of them are based on SVMs that are non-probabilistic classifiers and these two restrictions together with associated scaling problems create the need for an efficient multi-kernel multi-class probabilistic classifier.

To that end, another related approach within the non-parametric Gaussian process (GP) methodology^{33,36} has been proposed by,¹⁷ where instead of kernel combination the integration of information is achieved via combination of the GP covariance functions. However, a first order approximation for inverting the composite kernel is needed to obtain estimates of the values for the kernel weights and the method is restricted to kernel-type covariance functions.

Proposed method

Having identified the problem area we offer a principled solution via a multi-class multi-kernel pattern recognition machine that is able to combine feature sets and kernel spaces in an informative manner while at the same time inferring their significance and predictive contribution. The proposed methodology is general and can also be employed outside the “kernel-domain”, i.e without the need to embed the features into Hilbert spaces, by allowing for combination of basis function expansions. That is useful in the case of multi-scale problems and wavelets.³ The combination of kernels or dictionaries of basis functions as an alternative to classifier combination offers the advantages of reduced computational requirements, the ability to learn the significance of the individual sources and an improved solution based on the inferred significance.

Our work further develops the work of,¹⁵ which we generalize to the multi-class case by employing a multinomial probit likelihood and introducing latent variables that give rise to efficient Metropolis-Hastings within Gibbs sampling from the parameter posterior distribution through the introduction of auxiliary variables as demonstrated in the seminal paper by¹ and further extended by.²⁰ This enables us to retain the efficient hierarchical Bayesian structure of our model and provides us with an elegant solution for the multiclass setting.

Furthermore, we bound the marginal likelihood or model evidence³¹ and derive the Variational Bayes (VB) approximation for the proposed model, providing a fast and efficient solution to accompany the standard MCMC aforementioned approach. We are able to combine kernels in a general way via linear, product or binary rules and learn the associated weights to infer the significance of the sources.

In a summary our contribution^b offers:

^bParts of this work have been submitted for publication, see⁸⁻¹⁰

- A hierarchical Bayesian model with probabilistic output predictions.
- An explicit multi-class multi-kernel classifier.
- The ability to infer knowledge on three levels (regressors, importance of information channels, parameters of kernels), within the training of the classifier.
- A general framework for combining different sources of information, not dependent on the use of kernels and on a specific kernel type.
- A multitude of combination rules from linear and binary to product ones.

1.2. Intuition and Motivation

The intuition behind kernel combination methods is to create the same type of information in all feature spaces and then to combine them instructively via an appropriate rule. The proposed approach, as can be seen from Fig. 1.1 for the protein fold prediction problem, is based on the ability to embed each object description via the kernel trick^{38,39} into a kernel space (Hilbert space). This produces a similarity measure between objects in every feature space and then, having a common measure, we can combine informatively these similarities onto a composite kernel space.

Hence now, a single multiclass kernel machine can operate on that composite space effectively “disregarding” the number of feature spaces used. Inference by Bayes theorem on our hierarchical multiclass model enables us to learn the significance of each source and their predictive power by the corresponding kernel weights β , to learn the regressors and the kernel parameters without resorting to *ad-hoc* ensemble learning, combination of binary classifiers or parameter tuning. It is also worth noting that the type of kernel employed does not need to be same across feature spaces nor of a specific form as long as it is a valid kernel function ensuring a positive semi-definite symmetric matrix.

1.3. Multinomial Probit Kernel Combination

Consider S sources of information; From each one we have input variables \mathbf{x}_n^s as D^s -dimensional vectors^c for $s = 1, \dots, S$ and corresponding

^cThe superscripts denote “function of” unless otherwise specified. Matrices are denoted by \mathbf{M} , vectors by \mathbf{m} and scalars by m .

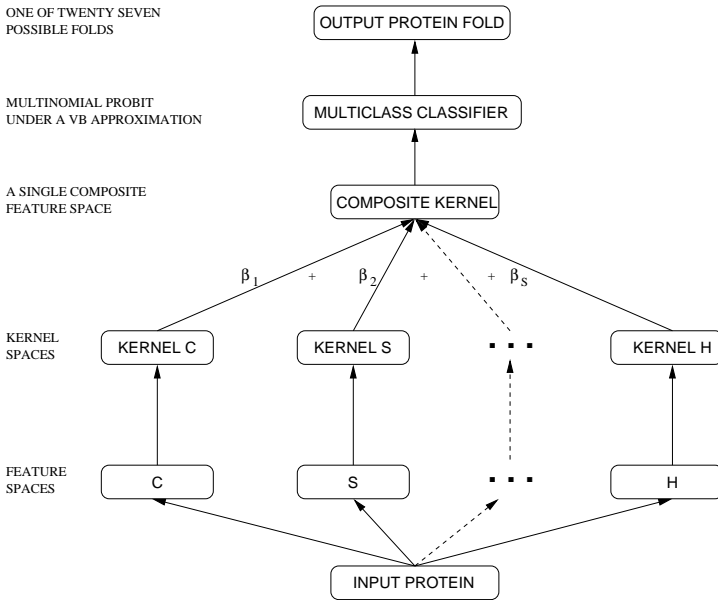


Figure 1.1. Diagrammatic representation of the linear kernel combination methodology for the specific problem of protein fold prediction. The original feature spaces are first embedded into kernels (Hilbert spaces) and then combined into a composite kernel where the multiclass kernel machine operates on.

real-valued target variables $t_n \in \{1, \dots, C\}$ for $n = 1, \dots, N$ where N is the number of observations and C the number of classes. By applying the *kernel trick* on the individual feature spaces created by the S sources we can define the $N \times N$ composite kernel as

$$\mathbf{K}^{\beta\Theta} = \sum_{s=1}^S \beta_s \mathbf{K}^{s\theta_s}$$

with each element analysed as

$$K^{\beta\Theta}(\mathbf{x}_i, \mathbf{x}_j) = \sum_{s=1}^S \beta_s K^{s\theta_s}(\mathbf{x}_i^s, \mathbf{x}_j^s)$$

where β is an $S \times 1$ column vector and Θ is an $S \times D^s$ matrix, describing the D^s -dimensional kernel parameters θ_s of all the base kernels \mathbf{K}^s . Now as we can see the composite kernel is a weighted summation^d of the base

^dWe present the more general linear kernel combination method first as our baseline approach.

kernels with β_s as the corresponding weight for each one. In the case where instead of kernels we employ the original feature spaces or expansions of them^e, we would still define the composite feature space as a weighted summation of the individual ones and proceed in a similar manner as below, with the appropriate dimensions for the corresponding sub-sets of regressors.

Following the standard approach for the multinomial probit by¹ we introduce auxiliary variables $\mathbf{Y} \in \mathcal{R}^{C \times N}$ and define the relationship between the auxiliary variable y_{cn} and the target variable t_n as

$$t_n = i \text{ if } y_{in} > y_{jn} \forall j \neq i \tag{1.1}$$

Now, the model response regressing on the variable y_{cn} with model parameters $\mathbf{W} \in \mathcal{R}^{C \times N}$ and employing a standardized normal noise model as in^{1,16} is given by

$$y_{cn} | \mathbf{w}_c, \mathbf{k}_n^{\beta\Theta} \sim \mathcal{N}_{y_{cn}} (\mathbf{w}_c \mathbf{k}_n^{\beta\Theta}, 1) \tag{1.2}$$

where $\mathcal{N}_x(m, v)$ denotes the normal distribution of x with mean m and variance v , \mathbf{W} and \mathbf{Y} are $C \times N$ matrices, \mathbf{w}_c is a $1 \times N$ row vector and $\mathbf{k}_n^{\beta\Theta}$ is an $N \times 1$ column vector from the n^{th} column of the composite kernel $\mathbf{K}^{\beta\Theta}$. Hence now, the multinomial likelihood, can be expressed as the following by simply marginalizing over the auxiliary variable \mathbf{y}_n and making use of relations 1.1 and 1.2:

$$\begin{aligned} P(t_n = i | \mathbf{W}, \mathbf{k}_n^{\beta\Theta}) &= \int P(t_n = i | \mathbf{y}_n) P(\mathbf{y}_n | \mathbf{W}, \mathbf{k}_n^{\beta\Theta}) d\mathbf{y}_n \\ &= \int \delta(y_{in} > y_{jn} \forall j \neq i) \prod_{c=1}^C \mathcal{N}_{y_{cn}} (\mathbf{w}_c \mathbf{k}_n^{\beta\Theta}, 1) d\mathbf{y}_n \\ &= \mathcal{E}_{p(u)} \left\{ \prod_{j \neq i} \Phi(u + (\mathbf{w}_i - \mathbf{w}_j) \mathbf{k}_n^{\beta\Theta}) \right\} \end{aligned} \tag{1.3}$$

where the expectation \mathcal{E} is taken with respect to the standardised normal distribution $p(u) = \mathcal{N}(0, 1)$. Hence, we can easily calculate the likelihood

^eConsider for example polynomial expansions.

by averaging the quantity inside the expectation for a certain number of random samples^f of u .

In the graphical model in Fig. 1.2, the conditional relation of the model parameters and associated hyper and hyper-hyper-parameters can be seen for the case of the *mean composite* kernel with the accompanied variations in Fig. 1.3 for the *binary* and *product composite* kernel. We place a product of zero mean Gaussian distributions on the regressors $\mathbf{W} \sim \prod_{c=1}^C \prod_{n=1}^N \mathcal{N}_{w_{cn}}(0, \zeta_{cn})$ with variance ζ_{cn} (described by the variable \mathbf{Z} in the graph) and a gamma distribution on the inverse of each scale with hyper-hyper-parameters τ, ν , reflecting our lack of prior knowledge and taking advantage of the conjugacy of these distributions.

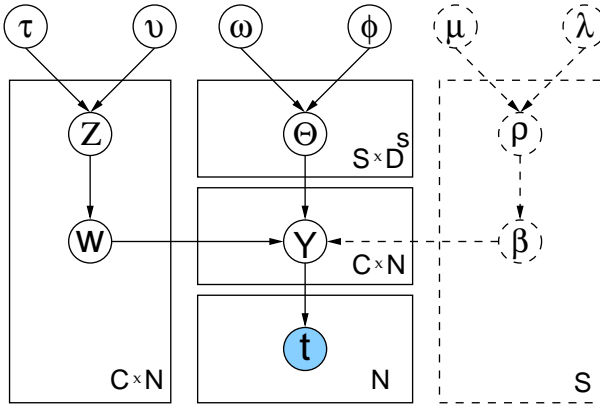


Figure 1.2. Plates diagram of the model for *linear composite* kernel.

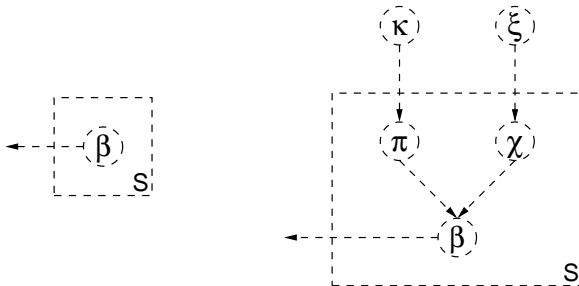


Figure 1.3. Modification for *binary composite* (left) and *product composite* kernel (right).

^fTypically 1000 samples have been employed which has been sufficient to approximate the expectation to a high degree of accuracy.

Furthermore, we place a gamma distribution on each kernel parameter since $\theta_{sd} \in \mathfrak{R}_+$. In the case of the *mean composite* kernel, a Dirichlet distribution with parameters ρ is placed on the combinatorial weights in order to satisfy the constraints imposed on the possible values which are defined on a simplex. A further gamma distribution is placed on each ρ_s with associated hyper-hyper-parameters μ, λ . The hyper-hyper-parameters $\Xi = \{\tau, v, \omega, \phi, \mu, \lambda, \kappa, \xi\}$ can be set by type-II maximum likelihood or set to uninformative values and the hyper and first level parameters $\Psi = \{\mathbf{Y}, \mathbf{W}, \beta, \rho, \Theta, \mathbf{Z}\}$ are sampled accordingly.

In the *product composite* kernel case we employ the right dashed plate in Fig. 1.3 which places a gamma distribution on the combinatorial weights β , that do not need to be defined on a simplex anymore, with an exponential hyper-prior distribution on each of the parameters π_s, χ_s .

Finally, in the *binary composite* kernel case we employ the left dashed plate in Fig. 1.3 which places a binomial distribution on each β_s with equal probability of being 1 or zero (unless prior knowledge says otherwise). The small size of the possible 2^S states of the β vector allows for their explicit consideration in the inference procedure and hence there is no need to place any hyper-prior distributions.

1.3.1. Markov chain Monte Carlo solution

Inference on the model parameters is performed via Bayes rule in order to obtain the posterior distributions. In that way we update the probabilities of parameter values based on the data as our prior distributions (prior beliefs) are multiplied by the likelihood (evidence from data). We present here the resulting posterior distributions which are adoptions, for our problem, of standard results in the statistics literature, for a full treatment see.^{12,18} The conditional posterior distribution for the regression parameters $p(\mathbf{W}|\mathbf{Y}, \mathbf{K}^{\beta\Theta}, \zeta) \propto p(\mathbf{Y}|\mathbf{W}, \mathbf{K}^{\beta\Theta}) p(\mathbf{W}|\zeta)$ is a product of Gaussian distributions $\prod_{c=1}^C \mathcal{N}_{\mathbf{w}_c}(\mathbf{m}_c, \mathbf{V}_c)$ where

$$\mathbf{m}_c = \mathbf{V}_c (\mathbf{K}^{\beta\Theta} \mathbf{y}_c^T) \text{ and } \mathbf{V}_c = (\mathbf{K}^{\beta\Theta} \mathbf{K}^{\beta\Theta} + \mathbf{Z}_c^{-1})^{-1}$$

\mathbf{Z}_c is a diagonal matrix with ζ_c in the main diagonal and \mathbf{y}_c is an $1 \times N$ vector of the auxiliary variables for a specific class c . The posterior $p(\mathbf{Y}|\mathbf{W}, \mathbf{K}^{\beta\Theta}, \mathbf{t}) \propto p(\mathbf{t}|\mathbf{Y}) p(\mathbf{Y}|\mathbf{W}, \mathbf{K}^{\beta\Theta})$ over the auxiliary variables is a product of N C -dimensional conically truncated Gaussians given by

$$\prod_{n=1}^N \delta(y_{in} > y_{jn} \forall j \neq i) \delta(t_n = i) \mathcal{N}_{\mathbf{y}_n}(\mathbf{W} \mathbf{k}_n^{\beta\Theta}, \mathbf{I})$$

Furthermore, the posterior distribution $p(\boldsymbol{\zeta}|\mathbf{W}, \tau, \nu) \propto p(\mathbf{W}|\boldsymbol{\zeta})p(\boldsymbol{\zeta}|\tau, \nu)$ over the variances $\boldsymbol{\zeta}$ is of the same form as the prior with updated parameters $\tau^* = \tau + \frac{1}{2}$ and $\nu^* = \nu + \frac{1}{2}w_{cn}^2$ and finally the combinatorial weights $\boldsymbol{\beta}$, the associated hyper-parameter $\boldsymbol{\rho}$ and the base kernel parameters Θ are inferred via three Metropolis-Hastings,¹⁹ MH hereafter, subsamplers with appropriate acceptance ratios.⁹

$$\text{where } \Phi(\boldsymbol{\rho}) = \frac{\Gamma\left(\sum_{s=1}^S \rho_s\right)}{\prod_{s=1}^S \Gamma(\rho_s)}$$

with the proposed move symbolised by $*$ and the current state with i .

In the case of the *binary* combination method, $\boldsymbol{\beta}$ becomes a binary vector switching base kernels on or off. The approach has been motivated by the work of²⁰ on covariate set uncertainty where they employed a MH subsampler to learn a binary covariate indicator vector switching features on or off. We modify the approach, as a *kernel set uncertainty* by inferring the binary vector with an extra Gibbs step that depends on the conditional distribution of the auxiliary variable \mathbf{Y} given $\boldsymbol{\beta}$ and marginalised over the model parameters \mathbf{W} .

The prior distribution over $\boldsymbol{\beta}$ is now a binomial distribution $\boldsymbol{\beta} \sim \prod_{s=1}^S \text{Bi}(\sigma_s)$ with $\sigma_s = 0.5$ reflecting our lack of prior knowledge. The conditional distribution which is the extra Gibbs step introduced, here for switching off kernels, $p(\beta_i = 0 | \boldsymbol{\beta}_{-i}, \mathbf{Y}, \mathbf{K}^{s\theta_s} \forall s \in \{1, \dots, S\})$ is given by

$$\frac{p(\mathbf{Y}|\beta_i = 0, \boldsymbol{\beta}_{-i}, \mathbf{K}^{s\theta_s} \forall s \in \{1, \dots, S\},) p(\beta_i = 0 | \boldsymbol{\beta}_{-i})}{\sum_{j=0}^1 p(\mathbf{Y}|\beta_i = j, \boldsymbol{\beta}_{-i}, \mathbf{K}^{s\theta_s} \forall s \in \{1, \dots, S\},) p(\beta_i = j | \boldsymbol{\beta}_{-i})}$$

where $\mathbf{K}^{s\theta_s} \forall s \in \{1, \dots, S\}$ are all the base kernels. The case for switching on kernels follows logically from the above.

Finally, the marginal likelihood term $p(\mathbf{Y}|\boldsymbol{\beta}, \mathbf{K}^{s\theta_s} \forall s \in \{1, \dots, S\})$, that the Gibbs step depends on, is derived based on standard procedures from the literature,¹² as

$$\prod_{c=1}^C (2\pi)^{-\frac{N}{2}} |\boldsymbol{\Omega}_c|^{-\frac{1}{2}} \exp\left\{-\frac{1}{2}\mathbf{y}_c \boldsymbol{\Omega}_c \mathbf{y}_c^T\right\}$$

where $\boldsymbol{\Omega}_c = \mathbf{I} + \mathbf{K}^{\beta\Theta} \mathbf{Z}_c^{-1} \mathbf{K}^{\beta\Theta}$

The *product composite* kernel employs two MH subsamplers to sample β , from a gamma distribution this time, and the hyper-parameters π, χ from exponential distributions. The kernel parameters Θ are inferred in all cases via an extra MH subsampler, see⁹ for details).

The MH subsamplers only need to sample once every step as the main Gibbs sampler, which always accepts the proposed move, will lead them towards convergence. The overall method in pseudo-algorithmic format can be seen in algorithm 1.1

Algorithm 1.1 Gibbs sampler

```

1: Initialize Hyper-hyper-parameters  $\tau, v, \omega, \phi, \mu, \lambda$ 
2: Sample hyper-parameters  $\zeta, \rho$ 
3: Sample parameters  $\mathbf{W}, \Theta, \beta, \mathbf{Y}$ 
4: Create train kernels
5: for Gibbs iterations do
6:   Update posterior distributions (Bayes rule)
7:   Sample hyper-parameters and parameters
8:   Update kernels (if needed)
9: end for
10: Discard Burn-in period samples
11: Create test kernels  $\mathbf{K}^*$  given  $\beta, \Theta$  posteriors
12: Monte Carlo estimation of Eq. 1.3 with  $\mathbf{K}^*$  given  $\mathbf{W}$  posterior

```

1.3.2. Variational Bayes approximation

An approximation to the exact solution for the multinomial probit composite kernel model can be offered, in similar manner to previous work by,¹⁶ via the variational methodology⁴ which offers a lower bound on the model evidence by using an ensemble of factored posteriors to approximate the joint parameter posterior distribution. The joint likelihood of the model⁸ is defined as $p(\mathbf{t}, \Psi | \mathbf{X}, \Xi) = p(\mathbf{t} | \mathbf{Y}) p(\mathbf{Y} | \mathbf{W}, \beta, \Theta) p(\mathbf{W} | \mathbf{Z}) p(\mathbf{Z} | \tau, v) p(\beta | \rho) p(\Theta | \omega, \phi) p(\rho | \mu, \lambda)$ and the factorable ensemble approximation of the required posterior is $p(\Psi | \Xi, \mathbf{X}, \mathbf{t}) \approx Q(\Psi) = Q(\mathbf{Y}) Q(\mathbf{W}) Q(\beta) Q(\Theta) Q(\mathbf{Z}) Q(\rho)$. We can bound the model evidence using Jensen's inequality

$$\log p(\mathbf{t}) \geq \mathcal{E}_{Q(\Psi)} \{ \log p(\mathbf{t}, \Psi | \Xi) \} - \mathcal{E}_{Q(\Psi)} \{ \log Q(\Psi) \} \quad (1.4)$$

and minimise it (the lower bound is given in⁹) with distributions of the form $Q(\Psi_i) \propto \exp(\mathcal{E}_{Q(\Psi_{-i})} \{ \log p(\mathbf{t}, \Psi | \Xi) \})$ where $Q(\Psi_{-i})$ is the factorable ensemble with the i^{th} component removed.

⁸The *linear composite* kernel is considered as an example. Modifications for the *binary* and *product composite* kernel are straightforward.

The resulting posterior distributions for the approximation are given below with full details of the derivations in.⁹ First, the approximate posterior over the auxiliary variables is given by

$$Q(\mathbf{Y}) \propto \prod_{n=1}^N \delta(y_{i,n} > y_{k,n} \forall k \neq i) \delta(t_n = i) \mathcal{N}_{\mathbf{y}_n} \left(\widetilde{\mathbf{W}} \mathbf{k}_n^{\beta \tilde{\Theta}}, \mathbf{I} \right) \quad (1.5)$$

which is a product of N C -dimensional conically truncated Gaussians. The shorthand tilde notation denotes posterior expectations in the usual manner, i.e. $f(\widetilde{\beta}) = \mathcal{E}_{Q(\beta)}\{f(\beta)\}$, and the posterior expectations for the auxiliary variable follow as

$$\widetilde{y}_{cn} = \widetilde{\mathbf{w}}_c \mathbf{k}_n^{\beta \tilde{\Theta}} - \frac{\mathcal{E}_{p(u)}\{\mathcal{N}_u(\widetilde{\mathbf{w}}_c \mathbf{k}_n^{\beta \tilde{\Theta}} - \widetilde{\mathbf{w}}_i \mathbf{k}_n^{\beta \tilde{\Theta}}, 1) \Phi_u^{n,i,c}\}}{\mathcal{E}_{p(u)}\{\Phi(u + \widetilde{\mathbf{w}}_i \mathbf{k}_n^{\beta \tilde{\Theta}} - \widetilde{\mathbf{w}}_c \mathbf{k}_n^{\beta \tilde{\Theta}}) \Phi_u^{n,i,c}\}} \quad (1.6)$$

$$\widetilde{y}_{in} = \widetilde{\mathbf{w}}_i \mathbf{k}_n^{\beta \tilde{\Theta}} - \left(\sum_{c \neq i} \widetilde{y}_{cn} - \widetilde{\mathbf{w}}_c \mathbf{k}_n^{\beta \tilde{\Theta}} \right) \quad (1.7)$$

where Φ is the standardized cumulative distribution function (CDF) and $\Phi_u^{n,i,c} = \prod_{j \neq i,c} \Phi(u + \widetilde{\mathbf{w}}_i \mathbf{k}_n^{\beta \tilde{\Theta}} - \widetilde{\mathbf{w}}_j \mathbf{k}_n^{\beta \tilde{\Theta}})$. Next, the approximate posterior for the regressors can be expressed as

$$Q(\mathbf{W}) \propto \prod_{c=1}^C \mathcal{N}_{\mathbf{w}_c} \left(\widetilde{\mathbf{y}}_c \mathbf{K}^{\beta \tilde{\Theta}} \mathbf{V}_c, \mathbf{V}_c \right) \quad (1.8)$$

where the covariance is defined as

$$\mathbf{V}_c = \left(\sum_{i=1}^S \sum_{j=1}^S \widetilde{\beta}_i \widetilde{\beta}_j \mathbf{K}^{i \tilde{\theta}_i} \mathbf{K}^{j \tilde{\theta}_j} + \left(\widetilde{\mathbf{Z}}_c \right)^{-1} \right)^{-1} \quad (1.9)$$

and $\widetilde{\mathbf{Z}}_c$ is a diagonal matrix of the expected variances $\widetilde{\zeta}_1 \dots \widetilde{\zeta}_N$ for each class. The associated posterior mean for the regressors is therefore $\widetilde{\mathbf{w}}_c = \widetilde{\mathbf{y}}_c \mathbf{K}^{\beta \tilde{\Theta}} \mathbf{V}_c$ and we can see the coupling between the auxiliary variable and regressor posterior expectation.

The approximate posterior for the variances \mathbf{Z} is an updated product of inverse-gamma distributions and the posterior mean is given by $2\tau + 1/2\nu + \widetilde{w}_{cn}^2$, for details see.¹² Finally, the approximate posteriors for the kernel parameters $Q(\Theta)$, the combinatorial weights $Q(\beta)$ and the associated hyper-prior parameters $Q(\rho)$, or $Q(\pi), Q(\chi)$ in the product composite

kernel case, can be obtained by importance sampling² in a similar manner to¹⁶ since no tractable analytical solution can be offered.

Having described the approximate posterior distributions of the parameters and hence obtained the posterior expectations we turn back to our original task of making class predictions \mathbf{t}^* for N_{test} new objects \mathbf{X}^* that are represented by S different information sources \mathbf{X}^{s*} embedded into Hilbert spaces as base kernels $\mathbf{K}^{*s\theta_s, \beta_s}$ and combined into a composite test kernel $\mathbf{K}^{*\Theta, \beta}$. The predictive distribution for a single new object \mathbf{x}^* is given by $p(t^* = c | \mathbf{x}^*, \mathbf{X}, \mathbf{t}) = \int p(t^* = c | \mathbf{y}^*) p(\mathbf{y}^* | \mathbf{x}^*, \mathbf{X}, \mathbf{t}) d\mathbf{y}^* = \int \delta_c^* p(\mathbf{y}^* | \mathbf{x}^*, \mathbf{X}, \mathbf{t}) d\mathbf{y}^*$ which ends up as

$$p(t^* = c | \mathbf{x}^*, \mathbf{X}, \mathbf{t}) = E_{p(u)} \left\{ \prod_{j \neq c} \Phi \left[\frac{1}{\widetilde{\nu}_j^*} \left(u \widetilde{\nu}_c^* + \widetilde{m}_c^* - \widetilde{m}_j^* \right) \right] \right\} \quad (1.10)$$

where,

for the general case of N_{test} objects, $\widetilde{\mathbf{m}}_c^* = \widetilde{\mathbf{y}}_c \mathbf{K} (\mathbf{K}^* \mathbf{K}^{*T} + \mathbf{V}_c^{-1})^{-1} \mathbf{K}^* \widetilde{\mathbf{V}}_c^*$ and $\widetilde{\mathbf{V}}_c^* = (\mathbf{I} + \mathbf{K}^{*T} \mathbf{V}_c \mathbf{K}^*)$ while we have dropped the notation for the dependance of the train $\mathbf{K} (N \times N)$ and test $\mathbf{K}^* (N \times N_{test})$ kernels on Θ, β for clarity. In algorithm 1.2 we summarize the VB approximation in a pseudo-algorithmic fashion.

Inference and intuition

Finally, the intuition behind the main model parameters is:

W The regressors indicate the weight with which data point n “votes” for class c . With appropriate sparsity-encuding priors on the distribution of their variance, a relevance vector machine⁴² version of our model can be implemented. Furthermore, in the case of imbalanced problems separate prior distributions on the variance of the regressors can be enforced to induce sparsity on the over-represented class and improve predictive performance.

β The combinatorial weights indicate the significance of individual sources and their predictive ability based on the model likelihood.

Θ The kernel parameters control the applied smoothing on each feature set and through them we can infer the significance of the original dimensions **D**.

Algorithm 1.2 Variational Bayes approximation

```

1: Initialize  $\Xi$ , sample  $\Psi$ , create  $\mathbf{K}_s | \beta_s, \theta_s$  and hence  $\mathbf{K} | \beta, \Theta$ 
2: while Lower Bound changing do
3:    $\tilde{\mathbf{w}}_c \leftarrow \tilde{\mathbf{y}}_c \mathbf{K} \mathbf{V}_c$ 
4:    $\tilde{y}_{cn} \leftarrow \tilde{\mathbf{w}}_c \mathbf{k}_n^{\tilde{\beta} \tilde{\Theta}} - \frac{\varepsilon_{p(u)} \{ \mathcal{N}_u(\tilde{\mathbf{w}}_c \mathbf{k}_n^{\tilde{\beta} \tilde{\Theta}} - \tilde{\mathbf{w}}_i \mathbf{k}_n^{\tilde{\beta} \tilde{\Theta}}, 1) \Phi_u^{n,i,c} \}}{\varepsilon_{p(u)} \{ \Phi(u + \tilde{\mathbf{w}}_i \mathbf{k}_n^{\tilde{\beta} \tilde{\Theta}} - \tilde{\mathbf{w}}_c \mathbf{k}_n^{\tilde{\beta} \tilde{\Theta}}) \Phi_u^{n,i,c} \}}$ 
5:    $\tilde{y}_{in} \leftarrow \tilde{\mathbf{w}}_i \mathbf{k}_n^{\tilde{\beta} \tilde{\Theta}} - \left( \sum_{j \neq i} \tilde{y}_{jn} - \tilde{\mathbf{w}}_j \mathbf{k}_n^{\tilde{\beta} \tilde{\Theta}} \right)$ 
6:    $\tilde{\zeta}_{cn} \leftarrow \frac{\tau + \frac{1}{2}}{v + \frac{1}{2} w_{cn}^2}$ 
7:    $\tilde{\rho}, \tilde{\beta}, \tilde{\Theta} \leftarrow \hat{\rho}, \hat{\beta}, \hat{\Theta} | \tilde{\mathbf{w}}_c, \tilde{\mathbf{y}}_n$  by importance sampling
8:   Update  $\mathbf{K} | \beta, \Theta$  and  $\mathbf{V}_c$ 
9: end while
10: Create composite test kernel  $\mathbf{K}^* | \beta, \Theta$ 
11:  $\tilde{\mathbf{V}}_c^* \leftarrow \left( \mathbf{I} + \mathbf{K}^{*T} \mathbf{V}_c \mathbf{K}^* \right)$ 
12:  $\tilde{\mathbf{m}}_c^* \leftarrow \tilde{\mathbf{y}}_c \mathbf{K} \left( \mathbf{K}^* \mathbf{K}^{*T} + \mathbf{V}_c^{-1} \right)^{-1} \mathbf{K}^* \tilde{\mathbf{V}}_c^*$ 
13: for  $n = 1$  to  $N_{test}$  do
14:   for  $c = 1$  to  $C$  do
15:     for  $i = 1$  to  $K$  Samples do
16:        $u_i \leftarrow \mathcal{N}(0, 1), \quad p_{cn}^i \leftarrow \prod_{j \neq c} \Phi \left[ \frac{1}{\nu_j^*} \left( u_i \tilde{\nu}_c^* + \tilde{m}_c^* - \tilde{m}_j^* \right) \right]$ 
17:     end for
18:   end for
19:    $P(t_n^* = c | \mathbf{x}_n^*, \mathbf{X}, \mathbf{t}) = \frac{1}{K} \sum_{i=1}^K p_{cn}^i$ 
20: end for

```

1.4. Experiments and Results

In this section we present experimental results from both the MCMC and the VB methods, a direct comparison between them and an examination of their differences with respect to the resulting posterior distributions of the parameters. We monitor convergence of the VB approximation via the progression of the lower bound and assume convergence when there is less than 0.1% increase on the bound or a maximum number of 100 iterations has been reached. All results are repeated over a number of random initializations in order to report statistical measures. Finally, throughout the different data sets employed, we use the kernel function that was found to perform better after an initial examination and in the case of radial basis function kernels we fix the kernel parameters to $1/D$ where D the dimensionality of the corresponding features.

We first compare the MCMC solution and the approximation on two synthetic data sets, then we consider some publicly available UCI data sets and then we report results from the VB approximation on two important problems arising in the bioinformatics research area. The reported CPU times are for a 1.6 GHz Intel based PC with 2Gb RAM running Matlab codes.

1.4.1. Proof of concept on an artificial data-set

In order to illustrate the performance of the VB approximation against the full Gibbs sampling solution, we employ two low dimensional datasets which enable us to visualise the decision boundaries and posterior distributions produced by either method. First we consider a linearly separable case in which we construct the dataset by fixing our regressors $\mathbf{W} \in \mathbb{R}^{C \times D}$, with $C = 3$ and $D = 3$, to known values and sample two dimensional covariates \mathbf{X} plus a constant term. In that way, by knowing the true values of our regressors, we can examine the accuracy of both the Gibbs posterior distribution and the approximate posterior estimate of the VB. In Fig. 1.4.1 the dataset together with the optimal decision boundaries constructed by the known regressor values can be seen.

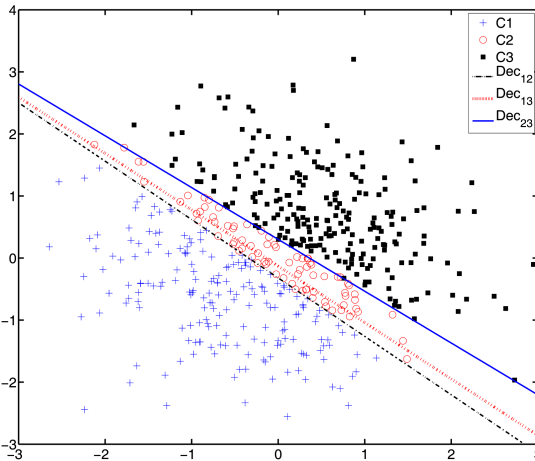


Figure 1.4. Linearly separable dataset with known regressors defining the decision boundaries. C_n denotes the members of class n and Dec_{ij} is the decision boundary between classes i and j .

In Figs. 1.5 and 1.6 we present the posterior distributions of one decision boundary's (Dec_{12}) slope and intercept based on both our obtained Gibbs samples and the approximate posterior of the regressors \mathbf{W} . As we can see, the variational approximation is in agreement with the mass of the Gibbs posterior and it successfully captures the pre-determined regressors values.

However, as it can be observed in Eq. 1.11 the approximation is over-confident in the prediction and produces a smaller covariance

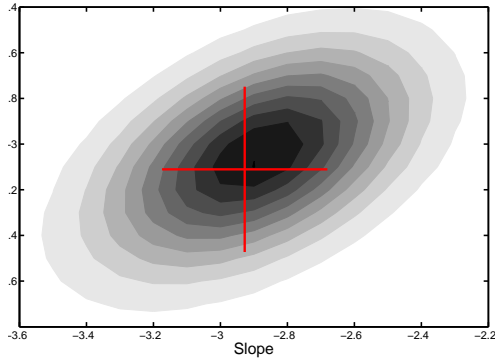


Figure 1.5. Gibbs posterior distribution of a decision boundary's (Dec_{12}) slope and intercept for a chain of 100,000 samples. The cross shows the original decision boundary employed to sample the dataset.

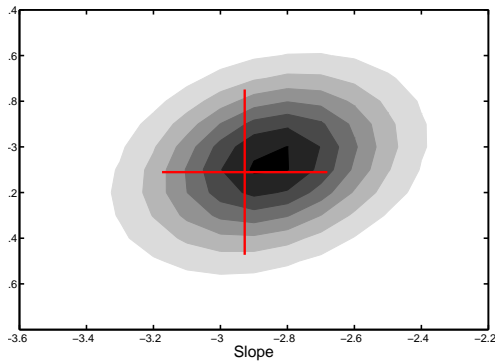


Figure 1.6. The variational approximate posterior distribution for the same case as above. Employing 100,000 samples from the approximate posterior of the regressors \mathbf{W} in order to estimate the approximate posterior of the slope and intercept.

for the posterior distribution as expected. Furthermore, the probability mass is concentrated in a very small area due to the very nature of VB approximation and similar mean field methods that make extreme “judgments” as they do not explore the posterior space by Markov chains.

$$C_{\text{Gibbs}} = \begin{bmatrix} 0.16 & 0.18 \\ 0.18 & 0.22 \end{bmatrix} \quad C_{\text{VB}} = \begin{bmatrix} 0.015 & 0.015 \\ 0.015 & 0.018 \end{bmatrix} \quad (1.11)$$

The second synthetic dataset we employ is a 4-dimensional 3-class dataset $\{\mathbf{X}, \mathbf{t}\}$ with $N = 400$, first described by,³³ which defines the first class as points in an ellipse $\alpha > x_1^2 + x_2^2 > \beta$, the second class as points below a line $\alpha x_1 + \beta x_2 < \gamma$ and the third class as points surrounding these areas, see Fig. 1.7.

We approach the problem by: 1) introducing a second order polynomial expansion on the original dataset $F(\mathbf{x}_n) = [1 \ x_{n1} \ x_{n2} \ x_{n1}^2 \ x_{n1}x_{n2} \ x_{n2}^2]$ while disregarding the uninformative dimensions x_3, x_4 , 2) modifying the dimensionality of our regressors \mathbf{W} to $C \times D$ and analogously the corresponding covariance and 3) substituting $F(\mathbf{X})$ for \mathbf{K} in our derived methodology. Due to our expansion we now have a $2-D$ decision plane that we can plot and a $6-D$ regressor \mathbf{w} per class. In Fig. 1.7 we plot the decision boundaries produced from the full Gibbs solution by averaging over the posterior parameters after 100,000 samples and in Fig. 1.8 the corresponding decision boundaries from the VB approximation after 100 iterations.

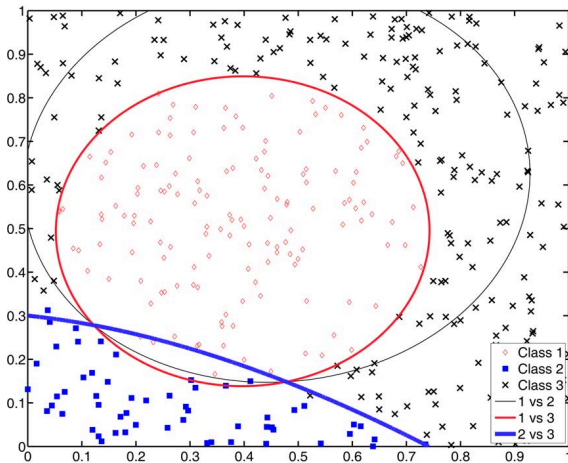


Figure 1.7. Decision boundaries from the Gibbs sampling solution.

As it can be seen, both the VB approximation and the MCMC solution produce similar decision boundaries leading to good classification performances - 2% error for both Gibbs and VB for the above decision boundaries and training size. However, the Gibbs sampler produces tighter boundaries due to the Markov Chain exploring the parameter posterior space more efficiently than the VB approximation.

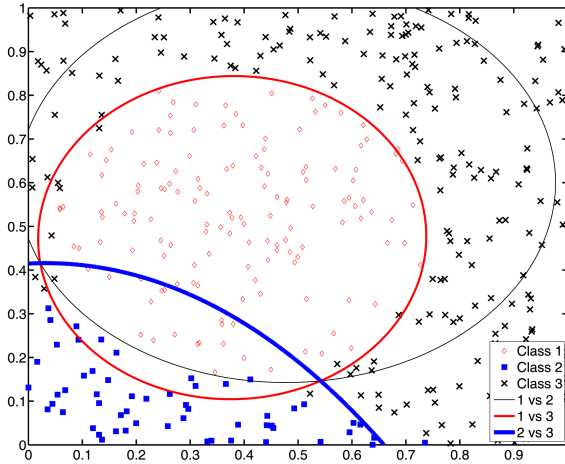


Figure 1.8. Decision boundaries from the VB approximation.

The corresponding training plus testing CPU times (sec) are given in Table 1.1 and it is straightforward to see the benefit of the approximation as an employable pattern recognition method.

Table 1.1. CPU times.

Gibbs	VB
41,720 (s)	120.3 (s)

1.4.2. UCI and handwritten numerals data-sets

To further assess the VB approximation of the proposed multinomial probit classifier, we explore a selection of UCI multinomial datasets. The performances are compared against reported results³² from well-known methods in the pattern recognition and machine learning literature. We employ an RBF (VB RBF), a 2^{nd} order polynomial (VB P) and a linear kernel (VB L) with the VB approximation and report 10-fold cross-validated (CV) error percentages, in Table 1.2, and CPU times, in Table 1.3, for a maximum of 50 VB iterations unless convergence has already occurred. The comparison with the K -nn and PK -nn is for standard implementations of these methods, see³² for details.

As we can see the VB approximation to the multinomial probit outperforms in most cases both the K -nn and PK -nn although not always

Table 1.2. 10-fold CV error % (mean \pm std) on standard UCI multinomial datasets.

Dataset	VB RBF	VB L	VB P	K-nn	PK-nn
Balance	8.8 \pm 3.6	12.2 \pm 4.2	7.0 \pm 3.3	11.5 \pm 3.0	10.2 \pm 3.0
Crabs	23.5 \pm 11.3	13.5 \pm 8.2	21.5 \pm 9.1	15.0 \pm 8.8	19.5 \pm 6.9
Glass	27.9 \pm 10.1	35.8 \pm 11.8	28.4 \pm 8.9	29.9 \pm 9.2	26.7 \pm 8.8
Iris	2.7 \pm 5.6	11.3 \pm 9.9	4.7 \pm 6.3	5.3 \pm 5.3	4.0 \pm 5.6
Soybean	6.5 \pm 10.6	6 \pm 9.7	4 \pm 8.4	14.50 \pm 16.7	4.5 \pm 9.6
Vehicle	25.6 \pm 4	29.7 \pm 3.3	26 \pm 6.1	36.3 \pm 5.2	37.2 \pm 4.5
Wine	4.5 \pm 5.1	2.8 \pm 4.7	1.1 \pm 2.3	3.9 \pm 3.8	3.4 \pm 2.89

Table 1.3. VB running times (seconds) for computing 10-fold cross-validation results.

Dataset	Balance	Crabs	Glass	Iris	Soybean	Vehicle	Wine
CPU time (s)	2,285	270	380	89	19	3,420	105

offering statistical significant improvements as the variance of the 10-fold CV errors is quite large in most cases.

In the following paragraphs we report results demonstrating the efficiency of our kernel combination approach when multiple sources of information are available. Comparisons are made against combination of classifiers and also between different ways to combine the kernels as we have described previously. In order to assess the above, we make use of a data-set that has multiple feature spaces describing the objects to be classified. It is a large $N = 2000$, multinomial $C = 10$ and “multi-featured” $S = 4$ UCI data-set, named “Multiple Features”. The objects are handwritten numerals, from 0 to 9, and the available features are the Fourier descriptors (FR), the Karhunen-Loève features (KL), the pixel averages (PX) and the Zernike moments (ZM).⁴¹ have previously reported results on this problem by combining classifiers but have employed a different test set which is not publicly available. Furthermore, we allow the rotation invariance property of the ZM features to cause problems in the distinction between digits 6 and 9. The hope is that the remaining feature spaces can compensate on the discrimination.

In Tables 1.4, 1.5 and 1.6, we report experimental results over 50 repeated trials where we have randomly selected 20 training and 20 testing objects from each class. For each trial we employ 1) A single classifier on each feature space, 2) the proposed classifier on the composite feature

Table 1.4. Classification percentage error (mean \pm std) of individual classifiers trained on each feature space.

Full MCMC Gibbs sampling - Single F.S			
FR	KL	PX	ZM
27.3 \pm 3.3	11.0 \pm 2.3	7.3 \pm 2	25.2 \pm 3

Table 1.5. Combinations of the individual classifiers based on four widely used rules: Product (Prod), Mean (Sum), Majority voting (Maj) and Maximum (Max).

Full MCMC Gibbs sampling - Comb. Classifiers			
Prod	Sum	Max	Maj
5.1 \pm 1.7	5.3 \pm 2	8.4 \pm 2.3	8.45 \pm 2.2

Table 1.6. Combination of feature spaces. VB approximation.

VB approx. - Comb. Kernels				
Bin	FixSum	WSum	FixProd	WProd
5.53 \pm 1.7	4.85 \pm 1.5	6.1 \pm 1.6	5.35 \pm 1.4	6.43 \pm 1.8

space. This allows us to examine the performance of combinations of classifiers versus combination of feature spaces. We report results from the VB approximation and in all cases we employ the multinomial probit kernel machine using a Gaussian kernel with fixed parameters.

As we can see from Table 1.4 and 1.5 the two best performing ensemble learning methods outperform all of the individual classifiers trained on separate feature spaces. With a p-value of $1.5e^{-07}$ between the Pixel classifier and the Product rule, it is a statistical significant difference. At the same time, all the kernel combination methods in Table 1.6 match the best performing classifier combination approaches, with a p-value of approximately 0.9 between the Product classifier combination rule and the linear (sum) kernel combination method.

Furthermore, the variants of our method that employ combinatorial weights offer the advantage of inferring the significance of the contributing sources of information. In Fig. 1.9, we can see that the pixel (PX) and zernike moments (ZM) feature spaces receive large weights and hence contribute significantly in the composite feature space. This is in accordance with our expectations, as the pixel feature space seems to be the best performing individual classifier and complementary predictive power mainly from the ZM channel is improving on that.

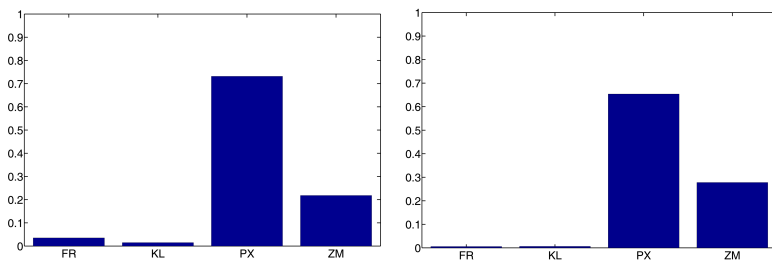


Figure 1.9. Typical combinatorial weights from Mean (left) and Product (right) composite kernel.

The results indicate that there is no benefit in the zero-one loss when weighted combinations of feature spaces are employed. This is in agreement with past work by²⁷ and¹⁷. The clear benefit however remains the ability to infer the relative significance of the sources and hence gain a better understanding of the problem.

1.4.3. *Protein fold recognition and remote homology detection*

Fold recognition

The original dataset from¹⁴ (based on SCOP PDB-40D) consists of 313 proteins for training and 385 proteins for testing with less than 35% sequence identity between any two proteins in the train and the test set. Furthermore, the extensions proposed by⁴⁰ exclude 4 proteins from the original dataset, namely proteins 2SCMC and 2GPS from the training set plus 2YHX_1 and 2YHX_2 from the test set, due to lack of sequence records.

Reported results are averaged over 20 (fold recognition) and 10 (RHD) randomly initialized trials in order to obtain statistical measures of accuracy and precision. Throughout this problem we have employed second order polynomial kernels for the global characteristics and inner product kernels for the local characteristics (SW) as they were found to provide a better embedding of the feature spaces.

First we examine the performance from individual feature spaces to gain an overall understanding of their predictive abilities. This however does not draw the complete picture as complementary information may be shared across sources achieving low performances. In Table 1.7 we present the mean percentage accuracy with standard deviations from our method (VBKC) together with the *best* ones reported by¹⁴ on the original dataset.

Table 1.7. Average individual F.S percentage accuracy.

Feature Space	VBKC	Ding and Dubchak
Amino Acid Composition (C)	51.2 \pm 0.5	44.9
Predicted Secondary Structure (S)	38.1 \pm 0.3	35.6
Hydrophobicity (H)	32.5 \pm 0.4	36.5
Polarity (P)	32.2 \pm 0.3	32.9
van der Waals volume (V)	32.8 \pm 0.3	35
Polarizability (Z)	33.2 \pm 0.4	32.9
PseAA $\lambda = 1$ (λ_1)	41.5 \pm 0.5	—
PseAA $\lambda = 4$ (λ_4)	41.5 \pm 0.4	—
PseAA $\lambda = 14$ (λ_{14})	38 \pm 0.2	—
PseAA $\lambda = 30$ (λ_{30})	32 \pm 0.2	—
SW with BLOSUM62 (SW ₁)	59.8 \pm 1.9	—
SW with PAM50 (SW ₂)	49 \pm 0.7	—

— Not employed in the Ding and Dubchak data-set.

Regarding the original features employed by¹⁴ we are in agreement with their observations as the best performing feature space, seems to be the amino acid composition (C). The $\lambda = 1$ and $\lambda = 4$ PseAA achieve the second best *global* individual performance and as the “step” λ increases further, the individual performances decrease. Although according to⁴⁰ the PseAA composition “has the same form as the conventional amino acid composition, but contains much more information” it seems at this stage that none of the PseAA is as predictive as the conventional amino acid composition. Furthermore, the local characteristics (SW) surprisingly outperform every global one and SW₁ achieves a higher accuracy than the best SVM-combinations proposed by.¹⁴ This is because although most of the proteins have less than 35% sequence similarity, this still seems to be an adequate similarity level to achieve a good accuracy.

In Table 1.8 we report the effect of sequentially adding the feature spaces in the order of,¹⁴ extending that to the addition of the PseAA compositions and finally adding the sequence similarity based features. We compare against the best performing SVM combination methodology as reported in¹⁴ and the ensemble method of.⁴⁰ As we can see in all the steps the proposed method outperforms the best reported accuracies and offers the current *state-of-the-art* in this data-set.

The best performances can be seen in Table 1.9 in comparison with the best ones reported in the cited past work. We achieve an improvement over both past methods while we employ a single multiclass kernel machine without resorting to ensemble learning techniques or combining multiple binary classifiers.

Table 1.8. Effect of F.S combination. % Accuracy reported.

Feature Spaces	VBKC	Ding & Dubchak (AvA)
C	51.2 ± 0.5	44.9
CS	55.7 ± 0.5	52.1
CSH	57.7 ± 0.6	56.0
CSHP	57.9 ± 0.9	56.5
CSHPV	58.1 ± 0.8	55.5
CSHPVZ	58.6 ± 1.1	53.9
CSHPVZ λ_1	60 ± 0.8	—
CSHPVZ $\lambda_1\lambda_4$	60.8 ± 1.1	—
CSHPVZ $\lambda_1\lambda_4\lambda_{14}$	61.5 ± 1.2	—
CSHPVZ $\lambda^1\lambda^4\lambda^{14}\lambda^{30}$	62.2 ± 1.3	—
CSHPVZ $\lambda^1\lambda^4\lambda^{14}\lambda^{30}SW_1$	66.4±0.8	—
CSHPVZ $\lambda^1\lambda^4\lambda^{14}\lambda^{30}SW_1SW_2$	68.1±1.2	—
		Shen & Chou
SHPVZ $\lambda_1\lambda_4\lambda_{14}\lambda_{30}$	61.0 ± 1.4	62.1

— Not employed in the Ding and Dubchak dataset.

Table 1.9. Best single run performances (% Accuracy).

Feature Spaces	Ding & Dubchak	Shen & Chou	VBKC
CSHPVZ	56	—	60.5
SHPVZ $\lambda_1\lambda_4\lambda_{14}\lambda_{30}$	—	62.1	63.5
CSHPVZ $\lambda_1\lambda_4\lambda_{14}\lambda_{30}$	—	—	63.9
CSHPVZ $\lambda^1\lambda^4\lambda^{14}\lambda^{30}SW_1SW_2$	—	—	70
No. of Classifiers	2,106	9	1

In Fig. 1.10 we plot a summary of the weights over 20 runs depicting the lower quartile, median, and upper quartile values.

As we can observe, the amino acid composition and the secondary structure are judged as more important, followed by the PseAA $\lambda = 1$. However, it is worth noting that by taking out the amino acid composition we have only a small loss in performance as we have seen in Table 1.8. These two observations suggest that the original amino acid (C) and the pseudo-ones (λ_i) carry redundant information. Furthermore, despite the individual accuracies of the SW features, they are not heavily weighted. This is because they depend solely on the sequence similarity between proteins and their quality of discriminative information is strongly related to which end of the 0-35% sequence similarity the two proteins will belong. In reality, for the real “twilight-zone” of low-homology proteins (much less than 35% similarity) such features have little effect by definition.

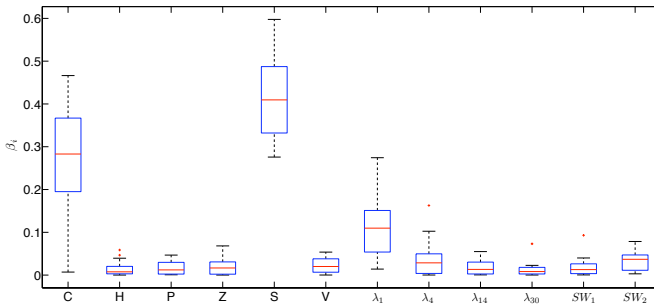


Figure 1.10. Combinatorial weights when all the feature spaces are employed.

Remote homology detection (RHD)

The SCOP 1.53 benchmark data-set^h as described in²⁸ is employed to simulate the RHD problem. It consists of 4,352 proteins belonging to one of 54 families and the positive training is performed on low-homologs while the positive testing on members of the same family. We consider four state-of-the-art string kernels, namely a *local alignment* (LA) kernel,³⁷ a *mismatch* (MM) kernel,²⁶ an *oligomer* kernel (Mono)²⁹ and a *pairwise* (PW) kernel,²⁸ taking the best performing case from each string kernel category as a separate informational source. We follow the above past works within the kernel machine paradigm by adding a class-dependent regularization parameter to the diagonal of the kernels to improve performance on this highly imbalanced problem.

The results from the combination of the string kernels are depicted in Table 1.10 together with the best previously reported results within the SVM methodology. We achieve a state-of-the-art performance via the combination of the kernels and match the overall best performing SVM method outperforming other string kernels. In Fig. 1.11 the number of families that achieve certain ROC scores is depicted in comparison with some of the best performing methods reported in the literature.

1.5. Discussion and Future Directions

In this contribution we offer a probabilistic multi-class multi-kernel machine that is able to operate simultaneously in multiple feature sets via a kernel

^hAvailable from <http://www.ccls.columbia.edu/complibio/svm-pairwise>

Table 1.10. ROC, ROC50 and median RFP averaged over 54 families.

Method	Mean ROC	Mean ROC50	Mean mRFP
VBKC	0.924	0.567	0.0661
SVM (SW)	0.896	0.464	0.0837
SVM (LA)	0.925	0.649	0.0541
SVM (MM)	0.872	0.400	0.0837
SVM (Mono)	0.919	0.508	0.0664

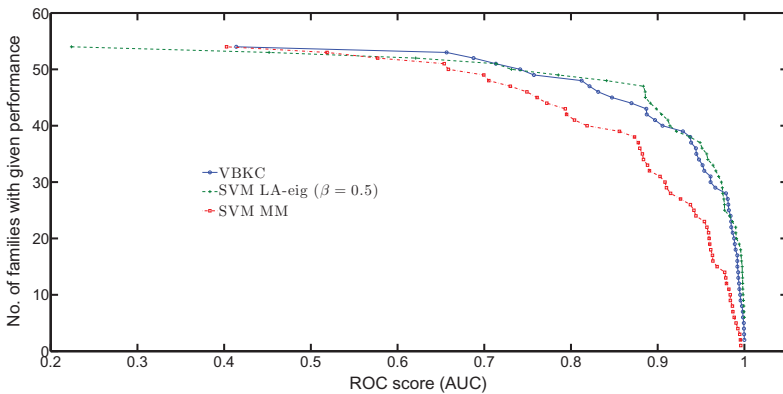


Figure 1.11. ROC score (AUC) distributions for the proposed string combination method and two state-of-the-art string kernels with SVMs.

combination methodology. Furthermore, we illustrate the capabilities of our method on two artificial data-sets, several benchmark UCI data-sets and finally on the challenging problems of protein fold and remote homology detection.

We provide the current state-of-the-art in the protein fold prediction problem and also achieve state-of-the-art performances in the majority of the considered data-sets, demonstrating the benefit of an explicit multi-class kernel machine for multi-featured problems. Furthermore, our methodology offers a significant reduction in computational resources, via the VB approximation, as it is based on a single classifier operating over a composite space which retains the dimensionality ($N \times N$) of any of the individual contributing feature spaces ($N \times N$). This, in contrast with the past work of employing thousands of binary classifiers or ensembles of individually trained classifiers is a significant improvement.

The computational complexity of the MCMC solution is $\sim \mathcal{O}(SN^3)$ per sample, with the VB approximation alleviating the sample size dependance. We are planning to offer the MAP solution to our model as a further approximate solution and to extend the method to an explicit multi-class multi-kernel RVM algorithm and also an informative vector machine (IVM)²⁴ adaptation which would offer further insight on the advantages and drawbacks of these frameworks.

Funding

NCR Financial Solutions Group Ltd. Scholarship to T.D; EPSRC Advanced Research Fellowship (EP/E052029/1) to M.A.G

Acknowledgment

The authors would like to acknowledge insightful discussions with Dr. David Leader and Dr. Rainer Breiting. The first author would like to acknowledge the support received from the Advanced Technology & Research Group within the NCR Financial Solutions Group Ltd company and especially the help and support of Dr. Gary Ross and Dr. Chao He.

References

1. Albert, J. and Chib, S. (1993). Bayesian analysis of binary and polychotomous response data, *Journal of the American Statistical Association* **88**, pp. 669–679.
2. Andrieu, C. (2003). An introduction to MCMC for machine learning, *Machine Learning* **50**, pp. 5–43.
3. Bai, L. and Shen, L. (2003). Combining wavelets with hmm for face recognition, in *Proceedings of the 23rd Artificial Intelligence Conference*.
4. Beal, M. J. (2003). *Variational Algorithms for approximate Bayesian Inference*, Ph.D. thesis, The Gatsby Computational Neuroscience Unit, University College London.
5. Bellman, R. (1961). *Adaptive Control Processes: A Guided Tour* (Princeton University Press).
6. Berger, J. O. (1985). *Statistical Decision Theory and Bayesian Analysis* (Springer Series in Statistics, Springer).
7. Bishop, C. M. (2006). *Pattern Recognition and Machine Learning* (Springer, New York, USA).
8. Damoulas, T. and Girolami, M. A. (2008a). A bayesian multi-class multi-kernel pattern recognition machine, *Pattern Recognition Letters Under Review*.

9. Damoulas, T. and Girolami, M. A. (2008b). Combining feature spaces for multinomial classification with bayesian inference, *Pattern Recognition Under Review*.
10. Damoulas, T. and Girolami, M. A. (2008c). Probabilistic multi-class multi-kernel learning: On protein fold recognition and remote homology detection, *Bioinformatics Under Review*.
11. de Freitas, N., Højén-Sørensen, P., Jordan, M. and Russell, S. (2001). Variational MCMC, in *Proceedings of the 17th conference in Uncertainty in Artificial Intelligence*.
12. Denison, D. G. T., Holmes, C. C., Mallick, B. K. and Smith, A. F. M. (2002). *Bayesian Methods for Nonlinear Classification and Regression* (Wiley Series in Probability and Statistics, West Sussex, UK).
13. Dietterich, T. G. (2000). Ensemble methods in machine learning, in *Proceedings of the 1st International Workshop on Multiple Classifier Systems*, pp. 1–15.
14. Ding, C. and Dubchak, I. (2001). Multi-class protein fold recognition using support vector machines and neural networks, *Bioinformatics* **17**, 4, pp. 349–358.
15. Girolami, M. and Rogers, S. (2005). Hierarchic Bayesian models for kernel learning, in *Proceedings of the 22nd International Conference on Machine Learning*, pp. 241–248.
16. Girolami, M. and Rogers, S. (2006). Variational Bayesian multinomial probit regression with Gaussian process priors, *Neural Computation* **18**, 8, pp. 1790–1817.
17. Girolami, M. and Zhong, M. (2007). Data integration for classification problems employing Gaussian process priors, in *Twentieth Annual Conference on Neural Information Processing Systems*.
18. Hastie, T., Tibshirani, R. and Friedman, J. (2001). *The Elements of Statistical Learning* (Springer Series in Statistics, Springer), ISBN 0-387-95284-5.
19. Hastings, W. K. (1970). Monte Carlo sampling methods using Markov chains and their applications, *Biometrika* **57**, pp. 97–109.
20. Holmes, C. C. and Held, L. (2006). Bayesian auxiliary variable models for binary and multinomial regression, *Bayesian Analysis* **1**, pp. 145–168.
21. Kittler, J., Hatef, M., Duin, R. P. W. and Matas, J. (1998). On combining classifiers, *IEEE Transactions on Pattern Analysis and Machine Intelligence* **20**, 3, pp. 226–239.
22. Kuncheva, L. I. (2004). *Combining Pattern Classifiers. Methods and Algorithms* (Wiley).
23. Lanckriet, G. R. G. (2004). Learning the kernel matrix with semidefinite programming, *Journal of Machine Learning Research* **5**, pp. 27–72.
24. Lawrence, N., Seeger, M. and Herbrich, R. (2003). Fast sparse gaussian process methods: The informative vector machine, in *Advances in Neural Information Processing Systems 15*, pp. 625–632.

25. Lee, W.-J., Verzakov, S. and Duin, R. P. (2007). Kernel combination versus classifier combination, in *7th International Workshop on Multiple Classifier Systems*.
26. Leslie, C. S., Eskin, E., Cohen, A., Weston, J. and Noble, W. S. (2004). Mismatch string kernels for discriminative protein classification, *Bioinformatics* **20**, 4, pp. 467–476.
27. Lewis, D. P., Jebara, T. and Noble, W. S. (2006). Nonstationary kernel combination, in *23rd International Conference on Machine Learning*.
28. Liao, L. and Noble, W. S. (2003). Combining pairwise sequence similarity and support vector machines for detecting remote protein evolutionary and structural relationships, *Journal of Computational Biology* **6**, 6, pp. 857–868.
29. Lingner, T. and Meinicke, P. (2004). Remote homology detection based on oligomer distances, *Bioinformatics* **22**, 18, pp. 2224–2231.
30. MacKay, D. (2003). *Information Theory, Inference and Learning Algorithms* (Cambridge University Press).
31. MacKay, D. J. C. (1992). The evidence framework applied to classification networks, *Neural Computation* **4**, 5, pp. 698–714.
32. Manocha, S. and Girolami, M. A. (2007). An empirical analysis of the probabilistic k-nearest neighbour classifier, *Pattern Recognition Letters*.
33. Neal, R. M. (1998). Regression and Classification using Gaussian process priors, *Bayesian Statistics* **6**, pp. 475–501.
34. Ong, C. S., Smola, A. J. and Williamson, R. C. (2005). Learning the kernel with hyperkernels, *Journal of Machine Learning Research* **6**, pp. 1043–1071.
35. Pavlidis, P., Weston, J., Cai, J. and Grundy, N. W. (2001). Gene functional classification from heterogenous data, in *5th Annual International Conference on Computational Molecular Biology*, pp. 242–248.
36. Rasmussen, C. E. and Williams, C. K. I. (2006). *Gaussian Processes for Machine Learning* (MIT Press, Cambridge, Massachusetts, USA), ISBN 0-262-18253-X.
37. Saigo, H., Vert, J.-P., Ueda, N. and Akutsu, T. (2004). Protein homology detection using string alignment kernels, *Bioinformatics* **20**, 11, pp. 1682–1689.
38. Schölkopf, B. and Smola, A. (2002). *Learning with Kernels* (The MIT Press).
39. Shawe-Taylor, J. and Cristianini, N. (2004). *Kernel Methods for Pattern Analysis* (Cambridge University Press).
40. Shen, H.-B. and Chou, K.-C. (2006). Ensemble classifier for protein fold pattern recognition, *Bioinformatics* **22**, 14, pp. 1717–1722.
41. Tax, D. M. J., van Breukelen, M., Duin, R. P. W. and Kittler, J. (2000). Combining multiple classifiers by averaging or by multiplying? *Pattern Recognition* **33**, pp. 1475–1485.
42. Tipping, M. E. (2001). Sparse Bayesian learning and the relevance vector machine, *Journal of Machine Learning Research* **1**, pp. 211–244.
43. Wolpert, D. H. (1992). Stacked generalization, *Neural Networks* **5**, 2, pp. 241–259.