Pattern Recognition 48 (2015) 2141–2150

Contents lists available at ScienceDirect

Pattern Recognition

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

An efficient radius-incorporated MKL algorithm for Alzheimer's disease prediction

Xinwang Liu [a], Luping Zhou [b,*], Lei Wang [b], Jian Zhang [c], Jianping Yin [a], Dinggang Shen [d,e,*]

[a] School of Computer, National University of Defense Technology, Changsha 410073, China
[b] School of Computer Science and Software Engineering, University of Wollongong, NSW 2522, Australia
[c] Faculty of Engineering and Information Technology, University of Technology, Sydney, NSW 2007, Australia
[d] The Department of Radiology and Biomedical Research Imaging Center (BRIC), University of North Carolina, Chapel Hill, NC 27599, USA
[e] Department of Brain and Cognitive Engineering, Korea University, Seoul, Republic of Korea

ARTICLE INFO

Article history:
Received 28 January 2013
Received in revised form
11 August 2014
Accepted 9 December 2014
Available online 17 December 2014

Keywords:
Multiple kernel learning
Radius-margin bound
Support vector machines
Alzheimer's disease
Neuroimaging

ABSTRACT

Integrating multi-source information has recently shown promising performance in predicting Alzheimer's disease (AD). Multiple kernel learning (MKL) plays an important role in this regard by learning the combination weights of a set of base kernels via the principle of margin maximisation. The latest research on MKL further incorporates the radius of minimum enclosing ball (MEB) of training data to improve the kernel learning performance. However, we observe that directly applying these radius-incorporated MKL algorithms to AD prediction tasks does not necessarily improve, and sometimes even deteriorate, the prediction accuracy. In this paper, we propose an improved radius-incorporated MKL algorithm for AD prediction. First, we redesign the objective function by approximating the radius of MEB with its upper bound, a linear function of the kernel weights. This approximation makes the resulting optimisation problem convex and globally solvable. Second, instead of using cross-validation, we model the regularisation parameter C of the SVM classifier as an extra kernel weight and automatically tune it in MKL. Third, we theoretically show that our algorithm can be reformulated into a similar form of the SimpleMKL algorithm and conveniently solved by the off-the-shelf packages. We discuss the factors that contribute to the improved performance and apply our algorithm to discriminate different clinic groups from the benchmark ADNI data set. As experimentally demonstrated, our algorithm can better utilise the radius information and achieve higher prediction accuracy than the comparable MKL methods in the literature. In addition, our algorithm demonstrates the highest computational efficiency among all the comparable methods.

1. Introduction

Pattern recognition techniques have been extensively applied to the analysis and diagnosis of medical diseases, and their effectiveness and significance have been well demonstrated in the literature [1–3]. In particular, accurate classification of people contracting a disease and the healthy population helps not only treatment but also early prevention. Therefore, developing better classification methods in this regard is highly desired. In this paper, we aim to develop a new pattern classification algorithm that can achieve improved classification performance when applied to Alzheimer's disease.

Alzheimer's disease (AD in short) is the most common neurodegenerative disease, covering 60–70% age-related dementia [4]. It is a fatal disease that worsens as it progresses. Mild cognitive impairment (MCI) is a precursor of AD. It is heterogeneous, with a conversion rate of 15% per year to AD [5]. Considering the immense cost on looking after AD patients, early identification of MCI and AD patients is of great significance. As a result, the following two classification tasks become important: (i) discriminating MCI patients from the healthy population; and (ii) discriminating the MCI patients who will convert to AD from those who will not. Since the two tasks can generally be viewed as predicting whether a person will develop towards or into AD, we call them collectively “AD prediction” for short in this paper.

Recent studies have demonstrated neuroimaging techniques as an important meanings for AD analysis [6,7]. For example, magnetic resonance imaging (MRI) shows grey matter morphometry, and

* Corresponding authors.
E-mail addresses: 1022xinwang.liu@gmail.com, xliu@uow.edu.au (X. Liu), lupingz@uow.edu.au (L. Zhou), leiw@uow.edu.au (L. Wang), jian.zhang@uts.edu.au (J. Zhang), jpyin@nudt.edu.cn (J. Yin), dgshen@med.unc.edu (D. Shen).

http://dx.doi.org/10.1016/j.patcog.2014.12.007
0031-3203/© 2014 Elsevier Ltd. All rights reserved.

Fluorodeoxyglucose (FDG) positron emission tomography (PET) shows metabolic activity. In this case, more effective AD prediction methods have been developed by combining the complementary information carried by these imaging modalities [8,9]. As seen in the recent literature, the combination methods can be performed at feature level [10,11,8,12–14] or classifier level [15]. A common practice of feature-level combination is to concatenate the features from different modalities into a long feature vector [10,11] and use it for classification. However, such concatenation usually requires proper normalisation of the features from different modalities. Otherwise, classification could be dominated by the features that have large variation but are not necessarily discriminative, leading to less satisfying classification performance.

In the past several years, multiple kernel learning (MKL) has shown superior performance to the methods using feature-level combination on AD prediction [15]. MKL is an important extension of support vector machines (SVM) [16] for handling multiple information sources. By predefining one (or multiple in general) "base" kernel function for each source, MKL aims to find the optimal linear combination weights of these kernels by maximising classification-performance-related criteria such as the margin of two classes. One of the representative algorithms is SimpleMKL [17]. It has been used for AD prediction by combining multiple modalities such as MRI, PET, and cerebrospinal fluid (CSF) parameters [12–14]. Due to its promising classification performance and solid theoretical foundation, SimpleMKL [17] is regarded as the state-of-the-art for AD prediction with multiple modalities.

Recent research [18–20] proposes to use more sophisticated criteria to optimise the kernel weights. In addition to the margin of two classes, these criteria consider the radius of minimum enclosing ball (MEB) of training data. The logic lies at that the radius affects the generalisation performance of SVM and it varies with the kernel weights. Hence, this radius shall be considered when seeking the optimal weight values. In the following, we call the MKL algorithms in [18–20] "radius-incorporated MKL" for short.

Our study observes that when applied to AD prediction, these radius-incorporated MKL algorithms do not necessarily improve, sometimes even deteriorate, the classification performance. By looking into this, we find that the tasks of AD prediction often have a small number of training samples and the involved classes are usually difficult to differentiate. Based on this observation, we hypothesise that the following two issues lead to the unsatisfying classification performance of these radius-incorporated MKL algorithms: (i) their objective functions are not convex. This usually leads to a locally (rather than globally) optimal solution. Unless the locally optimal solution is close enough to the (unknown in practice) globally optimal solution, the kernel weights will not be properly optimised; (ii) essentially as an SVM classifier, MKL also needs to tune the regularisation parameter C to attain good classification. The above radius-incorporated MKL algorithms employ multi-fold cross-validation technique to tune C. Nevertheless, when the number of training samples is small, this technique will become less reliable and may select an inappropriate value for C. Such a selection could lead to poor classification performance, especially when the classes are difficult to separate, as in the tasks of AD prediction.

To address the above two issues, we propose an improved radius-incorporated MKL algorithm. Firstly, to address the issue of non-convexity, we employ an approximation to the radius of MEB in our objective function, rather than directly using the radius as existing algorithms. This approximation can be shown as a linear function of the kernel weights. This makes our objective function convex and a globally optimal solution is therefore guaranteed, as proved in this paper. Also, we discuss the relationship between this approximation and the original radius to give a theoretical support for using this approximation. Secondly, to address the issue of tuning the parameter C, we do not use cross-validation. Instead, we define an extra dummy base kernel and relate C to the weight of this kernel. In doing so, C can be tuned with the other weights in MKL, and this mitigates the reliability issue of cross-validation in the case of small sample. This trick of tuning C has been used for model selection of SVM [21,22] and kernel learning [23]. However, it has not been integrated into the radius-incorporated MKL algorithms, and we find that tuning C in this way can effectively help improving the classification performance of MKL on the tasks of AD prediction.

In addition, the radius-incorporated MKL algorithm proposed in this paper brings computational advantage. Our objective function can be transformed into a form similar to that in the SimpleMKL. This allows our algorithm to be readily implemented by existing software packages. This merit does not apply to existing radius-incorporated MKL algorithms, which need more sophisticated optimisation algorithms. Also, as mentioned above, our algorithm tunes the parameter C via optimisation instead of timing-consuming cross-validation. These factors contribute to the higher computational efficiency, which will be experimentally demonstrated.

Experimental studies are conducted on 11 UCI machine learning benchmark data sets and three AD prediction tasks. We compare our algorithm with a set of state-of-the-art MKL algorithms, including unweighted average MKL, SimpleMKL [17], radius-incorporated algorithms in [18–20], and non-sparse MKL algorithms [24]. As will be demonstrated, our algorithm can achieve better classification performance on AD prediction tasks and higher computational efficiency than existing algorithms in comparison.

The rest of this paper is organised as follows. The background on MKL is reviewed in Section 2. In Section 3, we develop our algorithm and analyse its properties. After that, two factors contributing to the improvement of our algorithm are discussed, and two additional algorithms are designed to experimentally verify this discussion. Section 5 reports our experimental study and the conclusion is drawn in Section 6.

2. Background

Let $\{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$ be a set of n training samples. $\mathbf{x}_i (i = 1, \ldots, n)$ is a d-dimensional column vector in a Euclidean space $\mathbb{R}^d$. Let y_i be the class label of $\mathbf{x}_i$, and its value is $+1$ or -1. Let $\phi(\cdot) : \mathbb{R}^d \rightarrow \mathcal{H}$ be a probably nonlinear mapping from $\mathbb{R}^d$ to a higher-dimensional feature space $\mathcal{H}$. A kernel function of $\mathbf{x}_i$ and $\mathbf{x}_j$ is defined as the inner product between $\phi(\mathbf{x}_i)$ and $\phi(\mathbf{x}_j)$ [25]. It is expressed as $k(\mathbf{x}_i, \mathbf{x}_j) = \phi(\mathbf{x}_i)^\top \phi(\mathbf{x}_j)$, where $\top$ denotes the transpose of a vector. The mapping $\phi(\cdot)$ usually cannot be explicitly computed.

2.1. Traditional MKL

As previously mentioned, MKL employs a set of base kernels. Let m be the number of base kernels, and the p-th kernel is denoted by $k_p(\cdot,\cdot)$,[1] where $p = 1, \ldots, m$. Accordingly, let $\phi_p(\cdot) : \mathbb{R}^d \rightarrow \mathcal{H}_p$ be a probably nonlinear mapping associated with the p-th base kernel, where $\mathcal{H}_p$ is the p-th feature space. It is known that $k_p(\cdot,\cdot) = \phi_p(\cdot)^\top \phi_p(\cdot)$ by definition. Let γ_p be the weight of $k_p(\cdot,\cdot)$ and let $\boldsymbol{\gamma} = (\gamma_1, \ldots, \gamma_m)^\top$ be an m-dimensional column vector. Let $k(\cdot,\cdot;\boldsymbol{\gamma})$ denote a linear combination of these base kernels and it is expressed as $k(\cdot,\cdot;\boldsymbol{\gamma}) = \sum_{p=1}^{m} \gamma_p k_p(\cdot,\cdot)$. Note that we explicitly show $\boldsymbol{\gamma}$ as a parameter of this kernel to emphasise its dependence on

[1] Note that the symbol $k_p(\cdot,\cdot)$ is used to emphasise the kernel "function", while $k_p(\mathbf{x}_i, \mathbf{x}_j)$ is used to emphasise the kernel function "value".

these kernel weights. In this paper we call $k(\cdot,\cdot;\boldsymbol{\gamma})$ the "final" kernel for short. Defining $\phi(\cdot;\boldsymbol{\gamma}) = [\sqrt{\gamma_1}\phi_1(\cdot)^\top, \ldots, \sqrt{\gamma_m}\phi_m(\cdot)^\top]^\top$, it is not difficult to see that

$$k(\cdot,\cdot;\boldsymbol{\gamma}) = \sum_{p=1}^{m} \gamma_p k_p(\cdot,\cdot) = \sum_{p=1}^{m} (\sqrt{\gamma_p}\phi_p(\cdot))^\top (\sqrt{\gamma_p}\phi_p(\cdot)) = \phi(\cdot;\boldsymbol{\gamma})^\top \phi(\cdot;\boldsymbol{\gamma}). \tag{1}$$

This result shows that *conceptually*, *MKL can be viewed as mapping a sample* $\mathbf{x}$ *onto a feature space* $\mathcal{H}(\boldsymbol{\gamma})$ *via* $\phi(\cdot;\boldsymbol{\gamma})$ *and using a single kernel* $k(\cdot,\cdot;\boldsymbol{\gamma})$. This concept will be frequently used to derive our algorithm.

As mentioned in Section 1, MKL aims to seek the optimal kernel weights $\boldsymbol{\gamma}$. Most of existing MKL algorithms [17,24,26] find the optimal $\boldsymbol{\gamma}$ by maximising the margin of two classes as

$$\min_{\boldsymbol{\gamma},\boldsymbol{\omega},b,\boldsymbol{\xi}} \frac{1}{2}\|\boldsymbol{\omega}\|^2 + C\sum_{i=1}^{n}\xi_i$$
$$\text{s.t.}\quad y_i(\boldsymbol{\omega}^\top \phi(\mathbf{x}_i;\boldsymbol{\gamma}) + b) \geq 1-\xi_i, \quad \xi_i \geq 0, \forall i, \quad \sum_{p=1}^{m}\gamma_p = 1, \ \gamma_p \geq 0, \ \forall p. \tag{2}$$

where $\boldsymbol{\omega}$ is the normal of the SVM separating hyperplane, b the bias of this hyperplane, and ξ_i the slack variable for sample $\mathbf{x}_i$. Two constrains ($\sum_{p=1}^{m}\gamma_p = 1$ and $\gamma_p \geq 0$) are imposed to ensure that (i) the final kernel $k(\cdot,\cdot,\boldsymbol{\gamma})$ will not become unbounded due to an arbitrarily large γ_p value; and (ii) $k(\cdot,\cdot,\boldsymbol{\gamma})$ will maintain its positive-definiteness.

2.2. The radius of the minimum enclosing ball (MEB)

The MEB is a ball enclosing training samples with the minimum radius. Following [25], this ball can be found by solving

$$\min_{R,\mathbf{c}} R^2; \quad \text{s.t. } \|\phi(\mathbf{x}_i;\boldsymbol{\gamma}) - \mathbf{c}\|^2 \leq R^2, \quad \forall i = 1,\ldots,n, \tag{3}$$

where $\mathbf{c}$ and R denote the centre and the radius, respectively. Since $\phi(\cdot;\boldsymbol{\gamma})$ is usually not explicitly known, this problem is solved in its dual form [27]. Let $\mathbf{K}(\boldsymbol{\gamma})$ be the kernel matrices for the final kernel. The dual problem is expressed as

$$R_0^2(\boldsymbol{\gamma}) = \left\{\max_{\boldsymbol{\beta}} [\boldsymbol{\beta}^\top \operatorname{diag}(\mathbf{K}(\boldsymbol{\gamma})) - \boldsymbol{\beta}^\top \mathbf{K}(\boldsymbol{\gamma})\boldsymbol{\beta}]; \ \text{s.t. } \boldsymbol{\beta}^\top \mathbf{1} = 1, \quad \mathbf{0} \leq \boldsymbol{\beta}\right\}, \tag{4}$$

where $\operatorname{diag}(\mathbf{K}(\boldsymbol{\gamma}))$ denotes a column vector consisting of the diagonal entries of $\mathbf{K}(\boldsymbol{\gamma})$, $\boldsymbol{\beta}$ is an n-dimensional column vector representing the dual variables, $\mathbf{1}$ denotes a column vector with all entries being "1", and $R_0(\boldsymbol{\gamma})$ denotes the radius of the found MEB. Eq. (4) indicates two things: (i) the maximisation problem within the curly brackets is the dual problem to solve; and (ii) the maximum objective function value is equivalent to the squared radius of the MEB [25]. Readers are referred to [25, Section 7.1] or Section 7.3 in the SVM tutorial[2] for the detailed derivation of the radius of MEB.

2.3. Existing radius-incorporated MKL

It is known that the generalisation performance of SVM can be unbiasedly estimated by the leave-one-out error (LOO) on a training sample set [28]. Also, the LOO error is upper bounded by the quantity R^2/ρ^2, which is known as the radius-margin bound (RMB) in the literature [22]. Since the margin ρ equals $1/\|\boldsymbol{\omega}\|$ in SVM, the bound is often expressed as $R^2\|\boldsymbol{\omega}\|^2$. To obtain a classifier with excellent generalisation performance, a small LOO error is desired, which in turn prefers small $\|\boldsymbol{\omega}\|^2$ and small R^2. Existing radius-incorporated MKL algorithms just implement this idea, in a variety of ways.

The algorithm in [18] minimises the following objective function:

$$\min_{\boldsymbol{\gamma},\boldsymbol{\omega},b,\boldsymbol{\xi}} \frac{1}{2}\|\boldsymbol{\omega}\|^2 + \frac{C}{\sum_{p=1}^{m}\gamma_p R_p^2}\sum_{i=1}^{n}\xi_i^2;$$
$$\text{s.t. } y_i(\boldsymbol{\omega}^\top \phi(\mathbf{x}_i;\boldsymbol{\gamma}) + b) \geq 1-\xi_i, \forall i; \quad \sum_{p=1}^{m}\gamma_p = 1, \ \gamma_p \geq 0, \ \forall p, \tag{5}$$

where R_p is the radius of the MEB in the space $\mathcal{H}_p$ corresponding to $k_p(\cdot,\cdot)$. Instead of explicitly computing the radius $R_0^2(\boldsymbol{\gamma})$ in Eq. (4), this algorithm computes R_p for each base kernel and uses its linear combination $\sum_{p=1}^{m}\gamma_p R_p^2$ to approximate $R_0^2(\boldsymbol{\gamma})$. This approximation avoids solving $R_0^2(\boldsymbol{\gamma})$ and therefore brings computational advantage. In this paper, this approximation will be adopted into our algorithm. By doing so, we can obtain a radius-incorporated MKL algorithm that is theoretically more elegant and computationally more efficient than the one developed in [18].

Another algorithm in [19] minimises the following objective function:

$$\min_{\boldsymbol{\gamma},\boldsymbol{\omega},b,\boldsymbol{\xi}} \frac{1}{2}R_0^2(\boldsymbol{\gamma})\|\boldsymbol{\omega}\|^2 + C\sum_{i=1}^{n}\xi_i;$$
$$\text{s.t. } y_i(\boldsymbol{\omega}^\top \phi(\mathbf{x}_i;\boldsymbol{\gamma}) + b) \geq 1-\xi_i, \quad \xi_i \geq 0, \forall i; \ \gamma_p \geq 0, \quad \forall p. \tag{6}$$

As seen, $R_0^2(\boldsymbol{\gamma})$ is explicitly computed here. To solve this problem, the work in [19] firstly converts the primal problem to its dual problem as

$$\max_{\boldsymbol{\alpha}} \left\{\boldsymbol{\alpha}^\top \mathbf{1} - \frac{1}{2R_0^2(\boldsymbol{\gamma})}(\boldsymbol{\alpha}\circ\mathbf{y})^\top \mathbf{K}(\boldsymbol{\gamma})(\boldsymbol{\alpha}\circ\mathbf{y})\right\};$$
$$\text{s.t. } \boldsymbol{\alpha}^\top \mathbf{y} = 0; \quad \mathbf{0} \leq \boldsymbol{\alpha} \leq C\mathbf{1}, \tag{7}$$

Due to the strong duality [27] of the primal problem, the minimum objective function value of the primal problem equals the maximum objective function value of the dual problem. Therefore, the work in [19] defines $\mathcal{J}(\boldsymbol{\gamma})$ as the maximum objective function value of Eq. (7) and reformulates Eq. (6) into

$$\min_{\boldsymbol{\gamma}} \mathcal{J}(\boldsymbol{\gamma}); \quad \text{s.t. } \gamma_p \geq 0, \quad \forall p, \tag{8}$$

where $\mathcal{J}(\boldsymbol{\gamma}) = \max_{\boldsymbol{\alpha}}\{\boldsymbol{\alpha}^\top \mathbf{1} - (1/2R_0^2(\boldsymbol{\gamma}))(\boldsymbol{\alpha}\circ\mathbf{y})^\top \mathbf{K}(\boldsymbol{\gamma})(\boldsymbol{\alpha}\circ\mathbf{y})\}$ subject to the constraints $\boldsymbol{\alpha}^\top \mathbf{y} = 0$ and $\mathbf{0} \leq \boldsymbol{\alpha} \leq C\mathbf{1}$.

The work in [19] proposes a tri-level optimisation process to solve the above problem. Specifically, at each iteration, (i) when a new set of kernel weights $\boldsymbol{\gamma}$ is obtained, $R_0^2(\boldsymbol{\gamma})$ will be updated by solving the quadratic programming (QP) problem in Eq. (4); (ii) The updated $R_0^2(\boldsymbol{\gamma})$ is combined into Eq. (7) to solve another QP problem to update $\boldsymbol{\alpha}$ and then a new value of $\mathcal{J}(\boldsymbol{\gamma})$ is obtained; (iii) After that, $\boldsymbol{\gamma}$ will be updated according to Eq. (8) and then go back to Step (i). The above procedure is repeated until convergence. As pointed out in [19], the optimisation problem in Eq. (8) is not convex with respect to $\boldsymbol{\gamma}$ and can only converge to a locally optimal solution.

Our algorithm will have an optimisation process similar to the above. However, it does not need to solve the QP problem in Step (i) due to adopting the approximation in [18]. As will be seen, our algorithm can achieve better performance in terms of classification and computation than [19]. Last but not least, comparing Eqs. (6) and (5) can find that the work in [19] does not impose the constraint $\sum_{p=1}^{m}\gamma_p = 1$ encountered at the end of Section 2.1. As highlighted in [19], this is an important advantage over [18] because this constraint is not necessarily an optimal setting. We shall let MKL automatically decide the scale of $\boldsymbol{\gamma}$ by utilising the radius information. Note that this desirable property is preserved in our algorithm.

[2] Available at: http://research.microsoft.com/pubs/67119/svmtutorial.pdf

3. The proposed algorithm

As indicated in Section 1, we believe that two issues, a non-convex objective function and the cross-validation-based regularisation parameter tuning, lead to the unsatisfying prediction performance of existing radius-incorporated MKL algorithms [18,19] on AD prediction tasks. Our improvement consists of two ideas. One is to approximate the radius of MEB with a linear function of the kernel weights, and the other is to jointly optimise the regularisation parameter with the kernel weights via MKL.

3.1. Approximation to the radius of MEB

We firstly show the relationship between the radius of MEB in the feature space $\mathcal{H}(\gamma)$ and a linear combination of the p radiuses in the feature space $\mathcal{H}_1, \ldots, \mathcal{H}_p$. It is this relationship who inspires and justifies our idea.

This relationship was observed in [18], as stated in Theorem 1. Recall that $\gamma_p (\gamma_p \ge 0)$ is the weight for the p-th base kernel; $R_0(\gamma)$ is the radius of MEB defined in Eq. (4); and R_p is the radius in $\mathcal{H}_p$.

Theorem 1. *It can be proved that* $R_0^2(\gamma) \le \sum_{p=1}^{m} \gamma_p R_p^2$.

For self-containedness, we have provided a more rigorous proof in the Appendix.

This theorem indicates that $\sum_{p=1}^{m} \gamma_p R_p^2$ is an upper bound of $R_0^2(\gamma)$. Note that R_p^2 can be pre-computed once the p base kernels are predefined, and it remains constant. Therefore, the only variables in $\sum_{p=1}^{m} \gamma_p R_p^2$ are the weights γ, and this upper bound is consequently a linear function of γ.

Inspired by this observation, we propose to approximate $R_0^2(\gamma)$ with this upper bound. It will bring forth the following benefits (as will be shown in the following parts): (i) our objective function can be proved to be an upper bound of the LOO error. This provides the theoretical justification for minimising our objective function to optimise γ; (ii) this approximation is pivotal for linking our algorithm to SimpleMKL, which allows it to be readily implemented by the off-the-shelf packages; and (iii) It makes our optimisation problem convex with respect to γ, and this greatly facilities the optimisation.

Since this approximation was also used in [18], we highlight the differences as follows: Firstly, that work cannot automatically handle the scaling issue of γ, as pointed out in [19]. As a result, an additional norm-constraint, $\sum_{p=1}^{m} \gamma_p = 1$, has to be imposed. However, which norm should be used is an issue itself. Our algorithm is free of this issue; secondly, our algorithm can automatically tune the parameter C, while cross-validation has to be used in [18]; lastly, our algorithm can achieve superior AD prediction performance to the algorithm in [18], as will be experimentally demonstrated.

3.2. The proposed radius-incorporated MKL algorithm (L2BRMKL)

The radius-margin bound is derived based on a hard-margin SVM [22]. Nevertheless, The classes in AD prediction cannot be well separated in general, and a soft-margin SVM is needed. To make this bound still applicable, we use a 2-normed soft-margin SVM[3] because it can be rewritten as a hard-margin SVM by slightly modifying its kernel matrix from $\mathbf{K}$ to $\mathbf{K}+\mathbf{I}/C$, where $\mathbf{I}$ is an identity matrix and C is the regularisation parameter [22,21].

To incorporate the radius information, we could directly minimise the radius-margin bound to optimise the weights γ. Applying the radius-margin bound gives the following optimisation problem (We assume that the 2-normed soft-margin SVM has been rewritten into a hard-margin one)

$$\min_{\gamma} \min_{\omega,b} \frac{1}{2} R_0^2(\gamma) \|\omega\|^2; \quad \text{s.t. } y_i(\omega^\top \phi(\mathbf{x}_i;\gamma)+b) \ge 1, \forall i, \quad \gamma_p \ge 0, \quad \forall p, \tag{9}$$

However, this optimisation shares the same problem of [19]. That is, a direct incorporation of the radius makes the problem non-convex, which is prone to being trapped into a local solution.

To handle this situation, we propose to approximate $R_0^2(\gamma)$ with the result in Theorem 1. Recall that the kernel matrix is modified from $\mathbf{K}(\gamma)$ to $\mathbf{K}(\gamma)+\mathbf{I}/C$. Now we define one more base kernel to account for this modification: γ_{m+1} is defined as $1/C$, and a dummy base kernel matrix is defined as the identity matrix $\mathbf{I}$. In this way, we can utilise MKL to jointly tune C.

Now, we formally propose the objective function of L2BRMKL as

$$\min_{\gamma} \min_{\omega,b} \frac{1}{2} \left(\sum_{p=1}^{m+1} \gamma_p R_p^2 \right) \|\omega\|^2; \quad \text{s.t. } y_i(\omega^\top \phi(\mathbf{x}_i;\gamma)+b) \ge 1, \forall i, \quad \gamma_p \ge 0, \quad \forall p. \tag{10}$$

Its properties are shown through the following propositions.

Proposition 1. *The objective function in Eq. (10) is an upper bound of the radius-margin bound in the form of* $\frac{1}{2} R_0^2(\gamma)\|\omega\|^2$.

Proof. By Theorem 1, we can obtain that $R_0^2(\gamma) \le \sum_{p=1}^{m+1} \gamma_p R_p^2$, and it leads to $\frac{1}{2} R_0^2(\gamma)\|\omega\|^2 \le \frac{1}{2}(\sum_{p=1}^{m+1} \gamma_p R_p^2)\|\omega\|^2$. This completes the proof. □

Proposition 1 indicates that by minimising this objective function, we can restrict the value of the radius-margin bound, which will in turn restrict the LOO error, an unbiased estimate of the generalisation error of SVM. This provides justification for our objective function.

In the following part, we show that our optimisation problem in Eq. (10) can be addressed via solving a convex optimisation problem. To this end, we rewrite the problem in Eq. (10) as

$$\min_{\gamma} \mathcal{J}(\gamma); \quad \text{s.t. } \gamma_p \ge 0, \quad \forall p, \tag{11}$$

where

$$\mathcal{J}(\gamma) = \left\{ \min_{\omega,b} \frac{1}{2} \left(\sum_{p=1}^{m+1} \gamma_p R_p^2 \right) \|\omega\|^2; \quad \text{s.t. } y_i(\omega^\top \phi(\mathbf{x}_i;\gamma)+b) \ge 1, \quad \forall i \right\}. \tag{12}$$

The following Theorem 2 shows that the problem in Eq. (11) can be reformulated into a form similar to SimpleMKL [17]. The mere but critical difference is that a radius-weighted norm-constraint is used, compared with an unweighted version in [17]. The significance of Theorem 2 lies at that (i) it reveals the connection of our algorithm with traditional MKL without using the radius information; (ii) it shows that our radius-incorporated MKL algorithm, which appears to be sophisticated, can essentially be reduced to a slightly changed traditional MKL. (iii) It suggests that our algorithm can be efficiently solved by the existing MKL software packages.

To prove Theorem 2, we first give Proposition 2 for the optimisation problem in Eq. (11). Its proof can be found in our previous work [20].

Proposition 2. *The objective function value* $\mathcal{J}(\gamma)$ *remains unchanged when* γ *is scaled to* $\tau\gamma$, *where the scale factor* τ *is any positive scalar. Also, the SVM decision function of the resulting MKL algorithm is not affected by* τ.

[3] In a 2-normed soft-margin SVM, the power of the slack variable ξ_i in the SVM object function is set as 2. The details can be found in [25, Chapter 7.2].

With Proposition 2, Theorem 2 shows that solving the optimisation in Eq. (11) can be converted to solving a related but simpler optimisation.

Theorem 2. *The optimal solution of the optimisation problem in Eq. (11), denoted as* $\boldsymbol{\gamma}^\star$, *can be written as* $\boldsymbol{\gamma}^\star = (\sum_{p=1}^{m+1}\gamma_p^\star R_p^2)\boldsymbol{\eta}^\star$, *where* $\boldsymbol{\eta}^\star$ *is the optimal solution of the following optimisation problem*

$$\min_{\boldsymbol{\eta}} \mathcal{J}(\boldsymbol{\eta}); \quad \text{s.t.} \sum_{p=1}^{m+1}\eta_p R_p^2 = 1, \eta_p \ge 0, \quad \forall p. \tag{13}$$

where

$$\mathcal{J}(\boldsymbol{\eta}) = \left\{ \min_{\tilde{\boldsymbol{\omega}},b} \frac{1}{2}\|\tilde{\boldsymbol{\omega}}\|^2; \quad \text{s.t. } y_i(\tilde{\boldsymbol{\omega}}^\top \phi(\mathbf{x}_i;\boldsymbol{\eta})+b) \ge 1, \forall i \right\}. \tag{14}$$

Also, for the SVM decision function of the resulting MKL algorithm, denoted by $f(\mathbf{x})$, *it can be proved that* $f(\mathbf{x}) = \tilde{\boldsymbol{\omega}}^\top\phi(\mathbf{x};\boldsymbol{\eta})+b = \boldsymbol{\omega}^\top\phi(\mathbf{x};\boldsymbol{\gamma})+b$.

Proof. Defining $\tilde{\boldsymbol{\omega}} := \sqrt{(\sum_{p=1}^{m+1}\gamma_p R_p^2)}\boldsymbol{\omega}$, Eq. (12) is rewritten as

$$\mathcal{J}(\boldsymbol{\gamma}) = \left\{ \min_{\tilde{\boldsymbol{\omega}},b} \frac{1}{2}\|\tilde{\boldsymbol{\omega}}\|^2; \quad \text{s.t. } y_i\left(\tilde{\boldsymbol{\omega}}^\top\phi\left(\mathbf{x}_i; \frac{\boldsymbol{\gamma}}{\sum_{p=1}^{m+1}\gamma_p R_p^2}\right)+b\right) \ge 1, \ \forall i. \right\} \tag{15}$$

Letting

$$\tau = \frac{1}{\sum_{p=1}^{m+1}\gamma_p R_p^2}$$

and $\boldsymbol{\eta} = \tau\boldsymbol{\gamma}$ (that is, $\eta_p = \tau\gamma_p, \forall p$), we obtain that

$$\sum_{p=1}^{m+1}\eta_p R_p^2 = \sum_{p=1}^{m+1}\tau\gamma_p R_p^2 = \tau\sum_{p=1}^{m+1}\gamma_p R_p^2 = 1. \tag{16}$$

The last equality is a direct result of the definition of τ. Also, applying Proposition 2, we can obtain $\mathcal{J}(\boldsymbol{\eta}) = \mathcal{J}(\tau\boldsymbol{\gamma}) = \mathcal{J}(\boldsymbol{\gamma})$. Hence, Eq. (15) can be rewritten with respect to $\boldsymbol{\eta}$ as

$$\mathcal{J}(\boldsymbol{\eta}) = \left\{ \min_{\tilde{\boldsymbol{\omega}},b} \frac{1}{2}\|\tilde{\boldsymbol{\omega}}\|^2; \quad \text{s.t. } y_i(\tilde{\boldsymbol{\omega}}^\top\phi(\mathbf{x}_i;\boldsymbol{\eta})+b) \ge 1, \forall i; \right.$$
$$\left. \sum_{p=1}^{m+1}\eta_p R_p^2 = 1, \quad \eta_p \ge 0, \quad \forall p. \right\}, \tag{17}$$

Because η_p is not a variable of this problem, the constraint on η_p can be moved out of the curly brackets and this gives the result in Eq. (14). Then, combing the constraints on η_p with $\min_{\boldsymbol{\eta}}\mathcal{J}(\boldsymbol{\eta})$ leads to the optimisation problem in Eq. (13) exactly. This proves the first part of this theorem.

We now prove the second part on SVM decision function. Note that $\phi(\mathbf{x};\boldsymbol{\eta})$ can be written as $[\sqrt{\eta_1}\phi_1(\mathbf{x})^\top, \dots, \sqrt{\eta_{m+1}}\phi_{m+1}(\mathbf{x})^\top]^\top$. We partition $\tilde{\boldsymbol{\omega}}$ in a similar manner as $\tilde{\boldsymbol{\omega}} = [\tilde{\boldsymbol{\omega}}_1^\top, \dots, \tilde{\boldsymbol{\omega}}_{m+1}^\top]^\top$, and then obtain that

$$f(\mathbf{x}) = \tilde{\boldsymbol{\omega}}^\top\phi(\mathbf{x};\boldsymbol{\eta})+b = \sum_{p=1}^{m+1}\tilde{\boldsymbol{\omega}}_p^\top\sqrt{\eta_p}\phi_p(\mathbf{x})+b = \sum_{p=1}^{m+1}\tilde{\boldsymbol{\omega}}_p^\top\sqrt{\tau\gamma_p}\phi_p(\mathbf{x})+b$$
$$= \sum_{p=1}^{m+1}\boldsymbol{\omega}_p^\top\sqrt{\gamma_p}\phi_p(\mathbf{x})+b = \boldsymbol{\omega}^\top\phi(\mathbf{x};\boldsymbol{\gamma})+b. \tag{18}$$

In the last two steps, we use the following facts: (i) $\tilde{\boldsymbol{\omega}} := \sqrt{(\sum_{p=1}^{m+1}\gamma_p R_p^2)}\boldsymbol{\omega} = \boldsymbol{\omega}/\sqrt{\tau}$; and (ii) $\phi(\mathbf{x};\boldsymbol{\gamma}) = [\sqrt{\gamma_1}\phi_1(\mathbf{x})^\top, \dots, \sqrt{\gamma_{m+1}}\phi_{m+1}(\mathbf{x})^\top]^\top$ and the partition that $\boldsymbol{\omega} = [\boldsymbol{\omega}_1^\top, \dots, \boldsymbol{\omega}_{m+1}^\top]^\top$. This completes the proof. □

As shown by Theorem 2, the solution of Eq. (11) can be obtained by solving Eq. (13). In the following part, we prove that the optimisation problem in Eq. (13) can be reformulated as a convex one. This will justify our claim that our algorithm can be addressed by solving a convex optimisation problem.

Proposition 3. *The optimisation problem in Eq. (13) is equivalent to*

$$\min_{\boldsymbol{\eta}} \min_{\widehat{\boldsymbol{\omega}},b} \frac{1}{2}\sum_{p=1}^{m+1}\frac{\|\widehat{\boldsymbol{\omega}}_p\|^2}{\eta_p}$$
$$\text{s.t.} \quad y_i\left(\sum_{p=1}^{m+1}\widehat{\boldsymbol{\omega}}_p^\top\phi_p(\mathbf{x}_i)+b\right) \ge 1, \forall i, \quad \sum_{p=1}^{m+1}\eta_p R_p^2 = 1, \quad \eta_p \ge 0, \quad \forall p, \tag{19}$$

which is jointly convex with respect to $\boldsymbol{\eta}$, $\widehat{\boldsymbol{\omega}}$ *and* b.

Proof. Let us define $\widehat{\boldsymbol{\omega}}_p := \sqrt{\eta_p}\tilde{\boldsymbol{\omega}}_p$. By substituting $\widehat{\boldsymbol{\omega}}_p$ into Eq. (14) and replacing $\mathcal{J}(\boldsymbol{\eta})$ in Eq. (13) with the result in Eq. (14), we obtain the optimisation problem in Eq. (19). Its objective function is a ratio of a quadratic function ($\|\widehat{\boldsymbol{\omega}}_p\|^2$) to a linear function ($\eta_p$). According to [27, Chapter 3.2.6, Example 3.18, p. 89], this function is convex. Also, because all the constraints are linear function of $\boldsymbol{\eta}$, $\widehat{\boldsymbol{\omega}}$ and b, the feasible domain of this optimisation is convex. Therefore, the problem in Eq. (19) is convex with respect to its variables. This completes the proof. □

Interestingly, we find that Eq. (19) has a form similar to the one in [17], with the only difference that the constraint $\sum_{p=1}^{m+1}\eta_p R_p^2 = 1$ is used in our algorithm while $\sum_{p=1}^{m}\eta_p = 1$ is used in [17]. The problem in Eq. (19) can be solved by any MKL packages sharing the same routine: updating the structural parameters of SVM, $\boldsymbol{\alpha}$, and the weights of base kernels, $\boldsymbol{\eta}$, alternately. Specifically, $\boldsymbol{\alpha}$ is updated with the current $\boldsymbol{\eta}$ by solving

$$\max_{\boldsymbol{\alpha}}\{\mathbf{1}^\top\boldsymbol{\alpha} - \tfrac{1}{2}(\boldsymbol{\alpha}\circ\mathbf{y})^\top\mathbf{K}(\boldsymbol{\eta})(\boldsymbol{\alpha}\circ\mathbf{y})\}; \quad \text{s.t. } \boldsymbol{\alpha}^\top\mathbf{y} = 0, \quad \boldsymbol{\alpha} \ge \mathbf{0}, \tag{20}$$

which is the dual problem of Eq. (14). Then, the weights $\boldsymbol{\eta}$ are updated with the obtained $\boldsymbol{\alpha}$ by using the reduced gradient descent method [17]. Specifically, the cost function for updating $\boldsymbol{\eta}$ is

$$\min_{\boldsymbol{\eta}} \mathbf{1}^\top\boldsymbol{\alpha}_0 - \frac{1}{2}(\boldsymbol{\alpha}_0\circ\mathbf{y})^\top\mathbf{K}(\boldsymbol{\eta})(\boldsymbol{\alpha}_0\circ\mathbf{y})$$
$$\text{s.t.} \sum_{p=1}^{m+1}\eta_p R_p^2 = 1, \quad \eta_p \ge 0, \quad \forall p, \tag{21}$$

where $\boldsymbol{\alpha}_0$ is obtained in the last iteration with fixed $\boldsymbol{\eta}$. Note that Eq. (21) is a constrained optimisation problem w.r.t $\boldsymbol{\eta}$. The positivity and equality constraints have to be maintained during the update of $\boldsymbol{\eta}$. Such problems can be effectively solved via the reduced gradient descent method [17]. This procedure repeats until convergence. Our algorithm is listed in Algorithm 1.

Algorithm 1. The proposed L2BRMKL.

```
1: Initialise $\boldsymbol{\eta}^0, \mathbf{K}_{m+1} = \mathbf{I}$ and $\eta_{m+1} = 1/C_0$.
2: Calculate $R_p^2 (p = 1, \dots, m+1)$ for each base kernel by
   following Eq. (3).
3: $i \leftarrow 0$
4: repeat
5:    Obtain $\boldsymbol{\alpha}^{i+1}$ by solving Eq. (20) with $\boldsymbol{\eta}^i$.
6:    Update $\boldsymbol{\eta}^{i+1}$ by solving Eq. (21) via the reduced gradient
   descent method [17] with $\boldsymbol{\alpha}^{i+1}$.
7: $i \leftarrow i+1$
8: until $\max\{|\eta_1^{i+1} - \eta_1^i|, \dots, |\eta_{m+1}^{i+1} - \eta_{m+1}^i|\} < 1e-4$
```

Applying Proposition 2 (or Theorem 2), we know that the SVM decision function produced by the problems in Eqs. (11) and (13) is the same. Also, according to Proposition 3, we can solve Eq. (13) by solving the equivalent problem in Eq. (19). Therefore, after we obtain the optimal $\boldsymbol{\alpha}^\star, b^\star$ and $\boldsymbol{\eta}^\star$ by solving Eq. (19), we can directly write the SVM decision function as

$$f(\mathbf{x})=\sum_{i=1}^{n}\alpha_i^{\star}y_i\sum_{p=1}^{m}\eta_p^{\star}k_p(\mathbf{x}_i,\mathbf{x})+b^{\star}, \tag{22}$$

and it will be used to classify test samples.

We close this subsection by highlighting the differences between the proposed L2BRMKL and our previous $\ell_2\,\mathrm{tr}\,\mathbf{S}_t$ MKL in [20], where the trace of a total scatter matrix in the feature space $\mathcal{H}(\boldsymbol{\gamma})$ is used to substitute $R_0^2(\boldsymbol{\gamma})$. Compared with $\ell_2\,\mathrm{tr}\,\mathbf{S}_t$ MKL, L2BRMKL has an objective function that can be proved as an upper bound related to the generalisation performance of the SVM classifier, as in Proposition 1. However, $\ell_2\,\mathrm{tr}\,\mathbf{S}_t$ MKL does not have this merit because the trace of the total scatter matrix is only a lower (rather than upper) bound of $R_0^2(\boldsymbol{\gamma})$ [29]. This difference is important in that it provides better theoretical justification for our algorithm. And we do observe the improvement brought by such a difference in the experimental study.

4. Discussion on the improvement

In this section, we discuss the potential aspects that contribute to the improvement achieved by our algorithm. Also, to better demonstrate how these aspects contribute, we develop two additional MKL algorithms in which only one aspect is improved while the other is kept unchanged.

4.1. Approximating the radius

To show the help of this aspect, we modify the objective function of the algorithm in [19], which directly incorporates the radius information by using $R_0^2(\boldsymbol{\gamma})$. Maintaining all the other settings in [19], we only replace $R_0^2(\boldsymbol{\gamma})$ with $\sum_{p=1}^{m}\gamma_p R_p^2$. Specifically, its objective function now becomes

$$\min_{\boldsymbol{\gamma},\boldsymbol{\omega},b,\boldsymbol{\xi}}\ \frac{1}{2}\left(\sum_{p=1}^{m}\gamma_p R_p^2\right)\|\boldsymbol{\omega}\|^2+C\sum_{i=1}^{n}\xi_i$$
$$\text{s.t.}\quad y_i(\boldsymbol{\omega}^\top\phi(\mathbf{x}_i;\boldsymbol{\gamma})+b)\ge 1-\xi_i,\ \xi_i\ge 0,\ \forall i,\qquad \sum_{p=1}^{m}\gamma_p=1,\ \gamma_p\ge 0,\quad \forall p. \tag{23}$$

Following the work in [19], the parameter C is chosen by cross-validation. Since this objective function is based on the SVM classifier with 1-normed soft margin, we call the resulting algorithm L1BRMKL+C. By comparing this algorithm with that in [19], we can see whether the approximation to the radius contributes to the improvement on the performance of MKL.

4.2. Automatically tuning the regularisation parameter C

We design another algorithm, termed as L2BRMKL+C, to investigate the benefit of automatically tuning C in MKL algorithms. L2BRMKL+C shares the same optimisation problem with the proposed L2BRMKL. They only differ in that L2BRMKL+C determines C by employing multi-fold cross-validation still. By comparing L2BRMKL with L2BRMKL+C, we can see whether automatically tuning C really helps.

5. Experimental results

This experiment aims to evaluate the proposed MKL algorithm, L2BRMKL, with respect to classification accuracy and computational efficiency. It is compared to a set of state-of-the-art MKL algorithms, including (i) the commonly used margin-only algorithm SimpleMKL [17]; (ii) three existing radius-incorporated algorithms RMKL [18], MBMKL [19], and $\ell_2\,\mathrm{tr}\,\mathbf{S}_t$ MKL [20]; (iii) recently developed non-sparse MKL algorithm NSMKL [24] which constrain the kernel weights with different norms; and (iv) unweighted MKL algorithm UWMKL that simply uses the average of all base kernels.

5.1. Data sets and experimental settings

We firstly use the 11 UCI machine learning benchmark data sets, which have been widely used to evaluate MKL algorithms [19,18,24]. Their names are listed in Table 2, and the data sets can be downloaded from the Internet.[4]

Every feature in these data sets is normalised to have zero mean and unit variance. To accumulate statistic, 30 training and test splits are created for each data set. For each split, 20% of samples in the data set are randomly selected as training data and the rest 80% is used for test. To predefine base kernels, we adopt four types of kernel functions, including Gaussian kernel $k(\mathbf{x}_i,\mathbf{x}_j)=\exp(-\|\mathbf{x}_i-\mathbf{x}_j\|^2/\sigma)$, Laplacian kernel $k(\mathbf{x}_i,\mathbf{x}_j)=\exp(-\|\mathbf{x}_i-\mathbf{x}_j\|/\sqrt{\sigma})$, inverse square distance kernel

$$k(\mathbf{x}_i,\mathbf{x}_j)=\frac{1}{\|\mathbf{x}_i-\mathbf{x}_j\|^2/\sigma+1},$$

and inverse distance kernel

$$k(\mathbf{x}_i,\mathbf{x}_j)=\frac{1}{\|\mathbf{x}_i-\mathbf{x}_j\|/\sqrt{\sigma}+1},$$

where σ denotes the kernel parameter. Let σ_0 stand for the average value of the pairwise Euclidean distance between samples in a training set. In this experiment, σ is set as $2^t\sigma_0$ with $t=-2,-1,0,1,2$, and employed for each kernel. In this way, we generate 20 (4×5) base kernels and use them in all the algorithms conducted on the 11 UCI data sets.

For each data set, an algorithm is trained and tested on the 30 training/test splits, and the average classification accuracy and standard deviation are reported. To conduct a rigorous comparison, the *paired Student's t-test* is performed. The *p*-value of this test represents the probability that the two sets of results in comparison come from the distributions with an equal mean. A *p*-value of 0.05 is considered statistically significant and used here.

In addition to the UCI data sets, an AD data set from the Alzheimer Disease Neuroimaging Initiative (ADNI) database is used. Each sample has 229 features from four data sources, including three CSF (cerebrospinal fluid) biomarkers, 63 left hippocampal shape features, 63 right hippocampal shape features and 100 regional grey matter volumes. As shown in Table 1, three tasks are defined to differentiate different clinic groups, including MCI vs. NC (normal control), PMCI (converters) vs. NC, and PMCI vs. SMCI (non-converters). In general, these clinic groups are largely overlapped with each other, making the prediction tasks challenging. MCI consists of two subgroups PMCI and SMCI. PMCI is more AD-like, while SMCI is more NC-like. Therefore, there is an increasing degree of difficulty to differentiate PMCI from NC, MCI from NC, and PMCI from SMCI. Especially, the last task, which tells the MCI converters (PMCI) from the non-converters (SMCI), is very challenging and also of great importance in AD prediction.

To predefine the base kernels, this experiment applies the above four types of kernel functions to each data source. For each source, five different kernel parameter σ values are set for each type of kernel in the same way as the UCI data sets. By doing so, 80 $(4\times 4\times 5)$ base kernels are created in total, with 20 base kernels for each data source. As seen from Table 1, the number of samples in these prediction tasks is generally small. To make a good use of these samples, we employ the LOO strategy to evaluate each MKL

[4] http://archive.ics.uci.edu/ml/datasets.html

algorithm. In this strategy, each sample is used as the test sample in turn to form a classification session. The classification results of all the sessions are pooled to obtain classification accuracy.

The proposed L2BRMKL algorithm can automatically tune C. For other algorithms, C has to be chosen by cross-validation. In this experiment, four-fold cross-validation is applied to a large range $[2^{-5}, 2^{-3}, \ldots, 2^{15}]$ to choose C. The experiment is conducted on a high-performance cluster, where each node has eight cores with 2.3 GHz CPU and 2 GB memory.

5.2. Results on UCI data sets

The classification results are listed in Table 2. For each data set, the highest accuracy and those whose differences from the highest

Table 1
Summary of the data sets used in the AD prediction experiments.

Data set	Instances		Features			
	# Positive	# Negative	# CSF biomarkers	Hippocampal shape		# Regional gray matter volumes
				# Left	# Right	
PMCI vs. NC	50	70	3	63	63	100
MCI vs. NC	121	70	3	63	63	100
PMCI vs. SMCI	50	71	3	63	63	100

Table 2
Comparison of classification accuracy (in percentage) obtained by different MKL algorithms on UCI data sets and the statistical test result. Boldface indicates the highest accuracy and those whose differences from the highest accuracy are not statistically significant (evaluated by paired Student's t-test with p-value ≥ 0.05). The three numbers in each cell represent the average classification accuracy, standard deviation and the p-value.

Data set	L2BRMKL **proposed**	ℓ2trStMKL [20]	SimpleMKL [17]	RMKL [18]	MBMKL [19]	Non-SparseMKL [24]		UWMKL
						$p=2$	$p=3$	
Coloncancer	**71.6 ± 6.6 1.00**	**71.5 ± 6.8 0.77**	67.9 ± 6.3 0.00	68.5 ± 7.0 0.00	68.3 ± 7.1 0.00	66.7 ± 5.6 0.00	66.0 ± 4.8 0.00	68.1 ± 5.2 0.00
Fourclass	**99.9 ± 0.0 1.00**	**99.9 ± 0.1 0.16**	**99.9 ± 0.0 1.00**	**99.9 ± 0.1 0.16**	**99.9 ± 0.0 1.00**	**99.8 ± 0.1 0.06**	99.9 ± 0.2 0.02	99.0 ± 0.5 0.00
Germannum	71.6 ± 1.0 0.00	71.5 ± 1.2 0.00	71.1 ± 1.3 0.00	71.5 ± 1.5 0.00	70.9 ± 1.2 0.00	**72.9 ± 1.5 1.00**	**72.7 ± 1.5 0.31**	72.2 ± 1.7 0.02
Heart	**82.9 ± 1.6 1.00**	81.5 ± 2.4 0.00	**82.2 ± 2.5 0.15**	80.9 ± 2.6 0.00	**82.7 ± 1.6 0.41**	81.4 ± 3.1 0.01	81.1 ± 2.9 0.00	79.5 ± 3.6 0.00
Ionosphere	**66.2 ± 3.0 0.52**	64.7 ± 1.1 0.00	65.1 ± 1.2 0.02	**65.3 ± 1.5 0.08**	**66.4 ± 3.3 1.00**	65.0 ± 1.2 0.02	65.0 ± 1.2 0.02	65.1 ± 1.2 0.03
Liver	**60.6 ± 2.3 0.22**	**59.9 ± 2.7 0.05**	59.6 ± 2.5 0.01	59.3 ± 2.5 0.00	**60.4 ± 2.6 0.16**	**61.2 ± 3.6 0.30**	**61.5 ± 3.9 0.69**	**61.7 ± 4.4 1.00**
Musk1	**83.1 ± 3.1 1.00**	76.7 ± 8.9 0.00	78.9 ± 4.5 0.00	80.4 ± 4.2 0.00	82.2 ± 3.7 0.00	53.8 ± 5.9 0.00	53.3 ± 6.9 0.00	51.2 ± 6.4 0.00
Sonar	**75.9 ± 3.2 1.00**	**75.7 ± 3.1 0.65**	73.2 ± 4.5 0.00	**74.9 ± 4.5 0.11**	74.6 ± 2.9 0.00	**74.2 ± 4.3 0.06**	**74.0 ± 5.6 0.12**	**73.9 ± 5.7 0.10**
Splice	68.7 ± 5.2 0.00	63.7 ± 3.9 0.00	64.7 ± 5.4 0.00	62.1 ± 4.4 0.00	**70.4 ± 4.5 1.00**	56.2 ± 4.4 0.00	55.8 ± 4.6 0.00	55.1 ± 4.5 0.00
Wdbc	**95.9 ± 0.9 1.00**	**95.8 ± 1.0 0.52**	95.4 ± 1.1 0.03	**95.8 ± 1.0 0.58**	**95.6 ± 1.1 0.15**	94.1 ± 2.1 0.00	94.2 ± 2.4 0.00	94.4 ± 2.3 0.00
Wpbc	**76.0 ± 0.3 0.23**	**75.8 ± 1.3 0.14**	**76.0 ± 0.5 0.21**	**76.1 ± 0.6 1.00**	**76.1 ± 0.2 0.60**	**75.6 ± 1.8 0.12**	**75.7 ± 1.4 0.16**	**75.3 ± 2.8 0.15**
Average	**77.5**	76.1	75.8	75.9	77.0	72.8	72.7	72.3
Win	**9**	6	3	5	7	5	4	3

one are not statistically significant are shown in bold. From this table, we can observe that (i) the proposed L2BRMKL and the existing radius-incorporated MKL algorithms [18–20] (with average accuracy of 77.5%, 75.9%, 77.0% and 76.1%) achieve overall better classification than the margin-based ones [17,24] (with average accuracy of 75.8% and 72.8% or 72.7%). This demonstrates the effectiveness of incorporating the radius information; (ii) among the radius-incorporated MKL algorithms, L2BRMKL attains the highest average accuracy 77.5% and wins on nine of the 11 data sets. This result initially validates its advantage and provides a basis to investigate its performance on AD prediction tasks in further.

5.3. Results on the tasks of AD prediction

In addition to classification accuracy, we adopt another criterion widely used in medical applications, i.e., Matthews Correlation Coefficient (MCC), to evaluate the proposed algorithm. MCC is defined as

$$\mathrm{MCC} = \frac{TP*TN - FP*FN}{\sqrt{(TP+FP)(TP+FN)(TN+FP)(TN+FN)}}, \tag{24}$$

where TP, TN, FP and FN represent the number of true positives, the number of true negatives, the number of false positives and the number of false negatives, respectively. As seen from the definition, MCC takes both "sensitivity" and "specificity" of the classification into account, and is a balanced measure of classification performance. The classification accuracy and MCC results are reported in Tables 3 and 4, respectively. As previous, the highest values are highlighted in bold for each task. Note that no standard deviation is reported here, because there is only one classification accuracy for each task when the LOO strategy is used. Also, due to this fact, the statistical test becomes inapplicable and no result is reported either.

From Table 3, we can see that L2BRMKL achieves the overall best classification performance in the three tasks. In the task of PMCI vs. NC, it obtains the equally best performance as $\ell_2 \operatorname{tr} \mathbf{S}_t$ MKL, RMKL and MBMKL. In the other two tasks, it outperforms all the other algorithms. Also, the advantage of L2BRMKL becomes more pronounced with the increasing degree of difficulty of the tasks. As seen, the most significant improvement is achieved on the task of PMCI vs. SMCI, where 3.3% improvement (75.2% vs. 71.9%) is gained over the second best one, SimpleMKL. In addition, for the easiest task of PMCI vs. NC, all the four radius-incorporated methods (L2BRMKL, RMKL, MBMKL, and $\ell_2 \operatorname{tr} \mathbf{S}_t$ MKL) perform better than the margin-only methods, indicating the advantage of incorporating radius information again. With the introduction of SMCI, the second and third tasks become more difficult. As observed, the performance of $\ell_2 \operatorname{tr} \mathbf{S}_t$ MKL, RMKL and MBMKL decreases significantly from 88.3% to 71.1%, 88.3% to 67.8%, and 88.3% to 69.4%, respectively. Although the performance of L2BRMKL also decreases due to the increased difficulty, its decrease is the smallest one. These results demonstrate the superiority of the proposed L2BRMKL on these AD prediction tasks. Also, this situation is further confirmed by the MCC values reported in Table 4, where the proposed L2BRMKL consistently shows the highest MCC values on the three AD prediction tasks.

To check whether the approximation to the radius (discussed in Section 4.1) contributes to the above improvement, we compare MBMKL in [19] and the L1BRMKL+C in Section 4. Recall that they

Table 5
Classification accuracy (in %) of L2BRMKL, L1BRMKL+C and MBMKL [19].

Data set	L2BRMKL proposed	L1BRMKL+C	MBMKL [19]
PMCI vs. NC	88.3	**90.8**	88.3
MCI vs. NC	**75.4**	71.2	72.3
PMCI vs. SMCI	**75.2**	73.6	69.4
Average	**79.6**	78.5	76.7

Table 6
Classification accuracy (in percentage) of L2BRMKL and L2BRMKL+C.

Data set	L2BRMKL proposed	L2BRMKL+C
PMCI vs. NC	**88.3**	87.5
MCI vs. NC	**75.4**	71.2
PMCI vs. SMCI	**75.2**	68.6
Average	**79.6**	75.8

Table 3
Classification accuracy (in percentage) of different MKL algorithms.

Data set	L2BRMKL proposed	ℓ2trStMKL [20]	SimpleMKL [17]	RMKL [18]	MBMKL [19]	Non-SparseMKL [24]		UWMKL
						$p=2$	$p=3$	
PMCI vs. NC	**88.3**	**88.3**	87.5	**88.3**	**88.3**	85.8	85.0	85.8
MCI vs. NC	**75.4**	74.4	72.3	69.6	72.3	74.8	73.8	73.3
PMCI vs. SMCI	**75.2**	71.1	71.9	67.8	69.4	70.3	67.8	67.8
Average	**79.6**	77.9	77.2	75.2	76.7	77.0	75.5	75.6
Win	**3**	1	0	1	1	0	0	0

Table 4
Comparison of MCC (in percentage) of different MKL algorithms.

Data set	L2BRMKL proposed	ℓ2trStMKL [20]	SimpleMKL [17]	RMKL [18]	MBMKL [24]	Non-SparseMKL [19]		UWMKL
						$p=2$	$p=3$	
PMCI vs. NC	**76.2**	**76.2**	74.9	76.0	**76.2**	71.2	69.7	71.0
MCI vs. NC	**45.9**	43.7	38.0	30.9	38.8	44.7	43.6	42.3
PMCI vs. SMCI	**48.2**	39.9	40.9	32.1	35.5	38.3	33.3	33.0
Average	**56.8**	53.3	51.3	46.3	50.2	51.4	48.9	48.8
Win	**3**	1	0	0	1	0	0	0

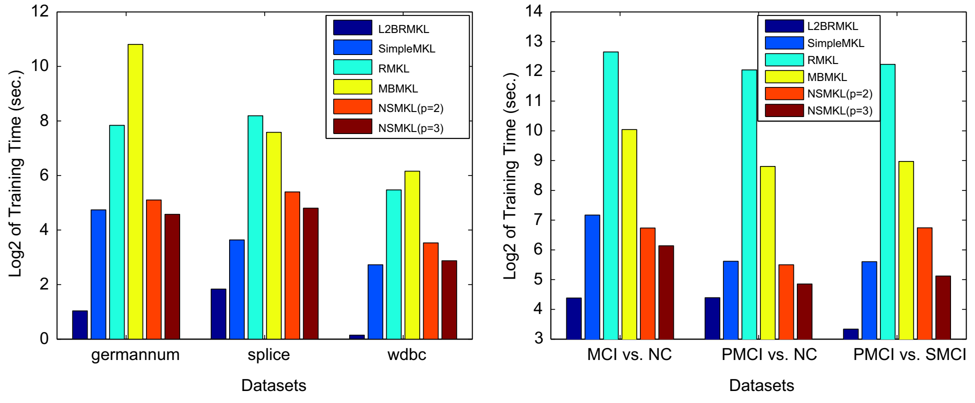

Fig. 1. Experimental comparison of timing results of different MKL algorithms, where L2BRMKL is the proposed method. (a) Timing result on three UCI data sets. (b): Timing result on three AD prediction tasks. (For interpretation of the references to color in this figure caption, the reader is referred to the web version of this article.)

only differ in the way to estimate the radius. The results are reported in Table 5. The performance of L2BRMKL is also quoted for reference.

From this table, we can see that L1BRMKL+C shows an overall better performance than MBMKL (78.5% vs. 76.7% by average), and achieves a clear improvement by 4.2% (73.6% vs. 69.4%) at the most difficult task of PMCI vs. SMCI. We conjecture that the inferiority of MBMKL to L1BMKL+C is due to the issue of numerical instability. Recall that MBMKL has a tri-level optimisation process, in which the radius of MEB is updated by solving a QP problem at each iteration. In this case, any numerical error occurring in solving this QP problem could adversely affect and entangle with the optimisation of structural parameter $\boldsymbol{\alpha}$ and in turn the kernel weights $\boldsymbol{\gamma}$. And such numerical errors could accumulate with the increasing number of iterations. A rigorous theoretical analysis of this numerical instability issue will be a good topic in our future work. At last, note that despite its advantage over MBMKL, L1BRMKL+C still performs overall worse than the proposed L2BRMKL.

To check the contribution of automatically tuning the regularisation parameter C, we compare L2BRMKL with the L2BRMKL+C in Section 4.2. The only difference between them is how C is tuned. As shown in Table 6, L2BRMKL consistently outperforms L2BRMKL+C. Also, the more difficult the task is, the more the improvement is attained. These results demonstrate the benefit of automatically tuning C on these classification tasks. The inferior performance of L2BRMKL+C is due to the following two factors: (i) the estimate of C obtained by cross-validation becomes unreliable when the number of training samples is small; and (ii) cross-validation can only examine a limited number of possible C values. These could result in a less appropriate C value and adversely affect the performance of L2BRMKL+C.

5.4. *Computational efficiency*

In this experiment, we first analyze the computational advantage of the proposed L2BRMKL and then conduct experimental comparison.

To facilitate the analysis, we use the commonly used SimpleMKL as a reference. In specific, we treat the computational cost of training SimpleMKL [17] on a set of samples with a preset regularisation parameter C as a unit, denoted by τ_0. Roughly speaking, under the same setting, the computational cost of L2BRMKL and NSMKL [24] will be at the order of τ_0, while the cost of RMKL [18] and MBMKL [19] will be higher than τ_0.

To choose a suitable value for C, the four existing algorithms employ multi-fold cross-validation. Let k be the number of folds and let s be the number of candidate C values tested in the cross-validation process. Then, we can know that for these four algorithms, their cost on cross-validation will be no less than the order of $ks\tau_0$[5]. Differently, by tuning C via the MKL process, the proposed L2BRMKL can maintain the cost at the order of τ_0, since only the number of kernel weights is slightly increased, from m to $m+1$. Also, among the four existing algorithms MBMKL [19] needs to compute the radius of MEB by solving a QP problem at each iteration. In contrast, the proposed L2BRMKL employs an approximation to the radius and therefore avoids such computation. This makes L2BRMKL more efficient than MBMKL in terms of integrating the radius information. Putting the above discussion together, we can see that L2BRMKL has the overall highest computational efficiency. Note that the above analysis does not depend on the machine, platform or language used to implement these algorithms.

As an experimental support to the above analysis, we compare the timing result of the above five MKL algorithms on three largest UCI data sets (*Germannum*, *Splice and Wdbc*) and the three AD prediction tasks. All the algorithms are implemented in Matlab, and no special measure is taken to optimise the speed of L2BRMKL. The timing result is the sum of cross-validation time and training time. The UWMKL algorithm is not included because it does not involve any learning procedure. The logarithm is applied to provide better illustration. As seen in Fig. 1(a), L2BRMKL (in black) is computationally much more efficient than the other MKL algorithms, especially when compared with the radius-incorporated ones, RMKL (in cyan) and MBMKL (in yellow). For example, on the data set of *Germannum*, the difference could reach the order of 10^3. Also, SimpleMKL and MSMKL generally show similar computational cost while RMKL and MBMKL are computationally most expensive. Fig. 1(b) shows the case for three AD prediction tasks, in which the computational advantage of the proposed L2BRMKL can still be seen. These timing results are well consistent with the above analysis.

6. Conclusion

Multiple kernel learning has become an effective method to predict AD by combining information from different sources. However, the recently developed radius-incorporated MKL algorithms do not give satisfactory performance on AD prediction tasks which are often difficult and only have a small number of training samples. To improve this situation, this paper proposes an improved radius-incorporated MKL algorithm to better handle the AD prediction tasks. Instead of rigidly computing the radius of

[5] Without loss of generality, we ignore the variation on training time due to that only $(k-1)/k$ of training samples are used in each training session of a k-fold cross-validation.

MEB, it approximates this radius with a linear combination of the radiuses pre-computed with each base kernel. Also, it absorbs the regularisation parameter into the MKL process and jointly optimises it with the other kernel combination weights. Through theoretical analysis, we discuss the connection of the proposed MKL algorithm to the well-known SimpleMKL algorithm and show that it can be readily solved. The effectiveness of the proposed algorithm is scrutinised and experimentally investigated on both pattern recognition benchmark data sets and three AD prediction tasks. As observed, it produces overall better classification performance and achieves higher computational efficiency.

The result in this paper also indicates that for the radius-incorporated MKL methods, how to compute the radius and design the objective function can significantly impact the kernel learning performance in practice. This raises interesting questions for both MKL research and its real applications, and they are worth exploring in the future work.

Conflict of interest

None declared.

Acknowledgement

Dr. Xinwang Liu was supported by the National Basic Research Program of China (973) under Grant no. 2014CB340303, the National Natural Science Foundation of China (project no. 61125201, 61303189, 61403405 and 61232016). Dr. Luping Zhou was supported by the UOW Vice Chancellor Research Grant. Dr. Dinggang Shen was supported in part by NIH grants EB006733, EB008374, EB009634, AG041721, MH100217, and AG042599.

Appendix A. Supplementary data

Supplementary data associated with this paper can be found in the online version at http://dx.doi.org/10.1016/j.patcog.2014.12.007.

References

[1] P.M. Rasmussen, L.K. Hansen, K.H. Madsen, N.W. Churchill, S.C. Strother, Model sparsity and brain pattern interpretation of classification models in neuroimaging, Pattern Recognit. 45 (6) (2012) 2085–2100.
[2] S.H. Park, S. Lee, I.D. Yun, S.U. Lee, Hierarchical MRF of globally consistent localized classifiers for 3D medical image segmentation, Pattern Recognit. 46 (9) (2013) 2408–2419.
[3] B. Caldairou, N. Passat, P.A. Habas, C. Studholme, F. Rousseau, A non-local fuzzy segmentation method: application to brain MRI, Pattern Recognit. 44 (9) (2011) 1916–1927.
[4] Alzheimer's Disease and Dementia: A Comparison of International Approaches, Technical Report, Report of the Social Committee on Aging, United States Senate, S. RES. 81, SEC. 17(d), March 2, 2011.
[5] J. Bischkopf, A. Busse, M.C. Angermeyer, Mild cognitive impairment-a review of prevalence, incidence and outcome according to current approaches, Acta Psychiatr. Scand. 106 (2002) 403–414.
[6] J. Ye, T. Wu, J. Li, K. Chen, Machine learning approaches for the neuroimaging study of Alzheimer's disease, IEEE Comput. 44 (4) (2011) 99–101.
[7] S. Duchesne, A. Caroli, C. Geroldi, C. Barillot, G.B. Frisoni, D.L. Collins, MRI-based automated computer classification of probable AD versus normal controls, IEEE TMI 27 (4) (2008) 509–520.
[8] J. Ye, K. Chen, T. Wu, J. Li, Z. Zhao, R. Patel, M. Bae, R. Janardan, H. Liu, G.E. Alexander, E. Reiman, Heterogeneous data fusion for Alzheimer's disease study, in: KDD, 2008, pp. 1025–1033.
[9] H. Wang, F. Nie, H. Huang, S. Kim, K. Nho, S.L. Risacher, A.J. Saykin, L. Shen, Identifying quantitative trait loci via group-sparse multitask regression and feature selection: an imaging genetics study of the ADNI cohort, Bioinformatics 28 (2) (2012) 229–237.
[10] Y. Fan, S.M. Resnick, X. Wu, C. Davatzikos, Structural and functional biomarkers of prodromal Alzheimer's disease: a high-dimensional pattern classification study, NeuroImage 41 (2) (2008) 277–285.
[11] C. Davatzikos, P. Bhatt, L.M. Shaw, K.N. Batmanghelich, J.Q. Trojanowski, Prediction of MCI to AD conversion, via MRI, CSF biomarkers, pattern classification, Neurobiol. Aging. 32 (12) (2011) 2322.e19-27, http://dx.doi.org/10.1016/j.neurobiolaging.2010.05.023.
[12] C. Hinrichs, V. Singh, G. Xu, S.C. Johnson, Predictive markers for AD in a multi-modality framework: an analysis of MCI progression in the ADNI population, Neuroimage 55 (2) (2011) 574–589.
[13] C. Hinrichs, V. Singh, G. Xu, S. Johnson, MKL for robust multi-modality AD classification, in: MICCAI, 2009, pp. 786–794.
[14] D. Zhang, Y. Wang, L. Zhou, H. Yuan, D. Shen, Multimodal classification of Alzheimer's disease and mild cognitive impairment, NeuroImage 55 (3) (2011) 856–867.
[15] Z. Dai, C. Yan, Z. Wang, J. Wang, M. Xia, K. Li, Y. He, Discriminative analysis of early Alzheimer's disease using multi-modal imaging and multi-level characterization with multi-classifier (M3), NeuroImage 59 (3) (2012) 2187–2195.
[16] N. Cristianini, J. Shawe-Taylor, An Introduction to Support Vector Machines: And Other Kernel-based Learning Methods, Cambridge University Press, New York, NY, USA, 2000, ISBN 0-521-78019-5.
[17] A. Rakotomamonjy, F. Bach, Y. Grandvalet, S. Canu, Simple MKL, J. Mach. Learn. Res. 9 (2008) 2491–2521.
[18] H. Do, A. Kalousis, A. Woznica, M. Hilario, Margin and radius based multiple kernel learning, in: Proceedings of the ECML, 2009, pp. 330–343.
[19] K. Gai, G. Chen, C. Zhang, Learning kernels with radiuses of minimum enclosing balls, in: NIPS, 2010, pp. 649–657.
[20] X. Liu, L. Wang, J. Yin, E. Zhu, J. Zhang, An efficient approach to integrating radius information into multiple kernel learning, IEEE Trans. Cybern. 43 (2) (2013) 557–569.
[21] S.S. Keerthi, Efficient turing of SVM hyperparameters using radius/margin bound and iterative algorithms, IEEE Trans. Neural Netw. 13 (5) (2002) 1225–1229.
[22] O. Chapelle, V. Vapnik, O. Bousquet, S. Mukherjee, Choosing multiple parameters for support vector machines, Mach. Learn. 46 (2002) 131–159.
[23] J. Ye, S. Ji, J. Chen, Multi-class discriminant kernel learning via convex programming, J. Mach. Learn. Res. 9 (2008) 719–758.
[24] Z. Xu, R. Jin, H. Yang, I. King, M. R. Lyu, Simple and efficient multiple kernel learning by group Lasso, in: Proceedings of the 27th ICML, 2010, pp. 1175–1182.
[25] J. Shawe-Taylor, N. Cristianini, Kernel Methods for Pattern Analysis, Cambridge University Press, Cambridge, 2004, ISBN 978-0-521-81397-6.
[26] M. Gönen, E. Alpaydın, Multiple kernel learning algorithms, J. Mach. Learn. Res. 12 (July) (2011) 2211–2268.
[27] S. Boyd, L. Vandenberghe, Convex Optimization, Cambridge University Press, New York, NY, USA, 2004, ISBN 0521833787.
[28] V. Vapnik, O. Chapelle, Bounds on error expectation for support vector machines, Neural Comput. 12 (9) (2000) 2013–2036, ISSN 0899-7667.
[29] L. Wang, Feature selection with kernel class separability, IEEE Trans. Pattern Anal. Mach. Intell. 30 (2008) 1534–1546.