Radius-margin ratio optimization for dot-product boolean kernel learning

Radius-margin ratio optimization for dot-product boolean kernel learning

Ivano Lauriola, Mirko Polato, and Fabio Aiolli

University of Padova - Department of Mathematics
Via Trieste, 63, 35121 Padova - Italy
ivanolauriola@gmail.com
{mpolato,aiolli}@math.unipd.it

Abstract. It is known that any dot-product kernel can be seen as a linear non-negative combination of homogeneous polynomial kernels. In this paper, we demonstrate that, under mild conditions, any dot-product kernel defined on binary valued data can be seen as a linear non-negative combination of boolean kernels, specifically, monotone conjunctive kernels (mC-kernels) with different degrees. We also propose a new radius-margin based multiple kernel learning (MKL) algorithm to learn the parameters of the combination. An empirical analysis of the MKL weights distribution shows that our method is able to give solutions which are more sparse and effective compared to the ones of state-of-the-art margin-based MKL methods. The empirical analysis have been performed on eleven UCI categorical datasets.

Keywords: Multiple Kernel Learning, radius-margin optimization, boolean kernels

1 Introduction

In the context of kernel machines, the choice of the kernel function is a key step to build good predictors. Kernel learning (KL), and the multiple kernel learning (MKL) paradigm in particular, aims at learning the best representation, i.e., the kernel function, directly from data. In the case of MKL, the used kernel is a combination of many base kernels. There exists several methods for combining kernels. In this paper, we consider only linear non-negative combinations of base kernels, in the form $\kappa(\mathbf{x},\mathbf{z}) = \sum_{r=0}^{R} \mu_r \kappa_r(\mathbf{x},\mathbf{z}),\ \mu_r \geq 0$.

Learning is usually supported by a validation step, where a user estimates the effectiveness of different kernels on a subset of training data, namely the validation set. More recently, alternative criteria have been proposed to estimate the goodness of a representation [4]. An important example of these strategies is the minimization of the radius-margin bound [2], that is the ratio between the radius of the minimum enclosing ball (MEB) and the margin observed on training data. In [5], for example, this strategy has been used for MKL optimization.

It is well known [3, 8] that any *dot-product kernel* (DPK) of the form $\kappa(\mathbf{x},\mathbf{z}) = f(\langle\mathbf{x},\mathbf{z}\rangle)$ can be seen as a *dot product polynomial* (DPP), that is a non-negative linear combination of *homogeneous polynomial kernels* (HP-kernels), i.e., $\kappa(\mathbf{x},\mathbf{z}) = \sum_{d=0}^{D} a_d \langle \mathbf{x},\mathbf{z}\rangle^d$, with appropriate coefficients $a_d \geq 0$. Recently, it has been shown that it is possible to generalize any DPK by making this combination non-parametric and by optimizing the coefficients a_d from data via a maximum margin MKL algorithm in both binary [3] and multiclass [6] contexts.

A first important contribution of the present paper is an extension of the above-mentioned result in the case of boolean input vectors $\mathbf{x},\mathbf{z} \in \{0,1\}^n$. Specifically, we demonstrate that any DPK defined on boolean vectors can be seen as a non-negative linear combination of *monotone conjunctive kernels* (mC-kernels) of different degrees. The mC-kernel of degree d basically counts the number of common true d-degree (positive) conjunctions of variables in the two input vectors and can be easily computed by a binomial coefficient $\kappa_{\wedge}^{d}(\mathbf{x},\mathbf{z}) = \binom{\langle \mathbf{x},\mathbf{z}\rangle}{d}$. The combination is referred to as *monotone conjunctive kernel polynomial* (mCKP).

Similarly to [3], here we propose to optimize the coefficients of a general mCKP via MKL. However, in our case, we propose a new gradient-descent method able to effectively minimize the exact radius-margin ratio. A similar kind of minimization have been proposed in [5] for MKL. However, in that work, different approximations were made to make the problem tractable.

We compare the proposed MKL algorithm (here dubbed RM-GD) in terms of AUC and the obtained radius-margin ratio on several categorical datasets, against two MKL baselines. Interestingly, we observed that, in almost every dataset, the distribution of the weights obtained by our MKL algorithm is very sparse and typically picked around two (one low degree and one high degree) mC-kernels. Hence, we refine our proposal by giving another simpler version of the algorithm that combines just one conjunctive kernel of a given degree with the identity matrix. This version has the advantage to be easily parallelizable.

2 Notation and background

Let $\mathbf{X} \in \{0,1\}^{l \times n}$ be the binary training matrix, and let $\mathbf{y} \in \{+1,-1\}^l$ be the vector of labels. We denote by $\kappa_{\wedge}^{d}(\mathbf{x},\mathbf{z}) = \binom{\langle \mathbf{x},\mathbf{z}\rangle}{d}$ and $\kappa_{HP}^{d}(\mathbf{x},\mathbf{z}) = \langle \mathbf{x},\mathbf{z}\rangle^d$ the d-degree monotone conjunctive kernel (mC-kernel) and the d-degree homogeneous polynomial kernel (HP-kernel) between $\mathbf{x}$ and $\mathbf{z}$, respectively.

The normalized version of a given kernel κ, here denoted $\tilde{\kappa}$, can also be considered. Note that, when needed, it can be easily computed by means of the well-known formula $\tilde{\kappa}(\mathbf{x},\mathbf{z}) = \kappa(\mathbf{x},\mathbf{z})/\sqrt{\kappa(\mathbf{x},\mathbf{x})\kappa(\mathbf{z},\mathbf{z})}$.

Given a training kernel matrix $\mathbf{K}$ such that $\mathbf{K}_{i,j} = \kappa(\mathbf{x}_i,\mathbf{x}_j)$, it can be shown that the margin obtained by a (hard-margin) SVM using that kernel can be computed as $\rho^2 = \min_{\gamma \in \Gamma} \boldsymbol{\gamma}^\top \mathbf{YKY}\boldsymbol{\gamma}$, where $\mathbf{Y}$ is the diagonal matrix of training labels $\mathbf{Y} = \mathit{diag}(\mathbf{y})$ and $\Gamma = \{\boldsymbol{\gamma} \in \mathbb{R}_{+}^{l} | \sum_{i:y_i=+1} \gamma_i = 1 \ \wedge \sum_{i:y_i=-1} \gamma_i = 1\}$.

Furthermore, it can be seen that, when a kernel is normalized, the radius of the MEB enclosing training data in feature space can be obtained by solving $R^2 = 1 - \min_{\boldsymbol{\alpha} \in \mathcal{A}} \boldsymbol{\alpha}^\top \mathbf{K}\boldsymbol{\alpha}$ where $\mathcal{A} = \{\boldsymbol{\alpha} \in \mathbb{R}_{+}^{l}, \sum_i \alpha_i = 1\}$.

Finally, given $\mathbf{x} \in \mathbb{R}^n$ and $\mathbf{p} \in \mathbb{N}_0^n$, the symbol $\mathbf{x}^{\mathbf{p}}$ will denote the product among variables exponentiated component-wise, that is $\mathbf{x}^{\mathbf{p}} = x_1^{p_1}, \ldots, x_n^{p_n}$.

3 DPKs as linear combinations of mC-kernels

The feature space of the HP-kernel of a given degree d is formed by all the monomials of degree d, each weighted by some coefficient. When the input vectors are binary, then many of these monomials collide in a single value, since the factors of the monomials x_i^p will have the same value for every $p \geq 1$. This observation allows us to give the following results concerning the relationship between HP-kernels and mC-kernels.

Theorem 1. *Given* $\mathbf{x}, \mathbf{z} \in \{0,1\}^n$*, then any HP-kernel can be decomposed as a finite non-negative linear combination of mC-kernels (a mCKP) of the form:*

$$\kappa^d_{HP}(\mathbf{x},\mathbf{z}) = \sum_{s=0}^{d} h(s,d)\ \kappa^s_{\wedge}(\mathbf{x},\mathbf{z}), \qquad h(s,d) \geq 0.$$

Proof. Given $\mathbf{x}, \mathbf{z} \in \{0,1\}^n$, by definition:

$$\kappa^s_{\wedge}(\mathbf{x},\mathbf{z}) = \binom{\langle \mathbf{x},\mathbf{z}\rangle}{s} = \sum_{\mathbf{b}\in\mathbb{B}_s} \mathbf{x}^{\mathbf{b}}\mathbf{z}^{\mathbf{b}} \tag{1}$$

where $\mathbb{B}_s \equiv \{\mathbf{b} \in \{0,1\}^n \mid \|\mathbf{b}\|_1 = s\}$. Moreover, we have:

$$\kappa^d_{HP}(\mathbf{x},\mathbf{z}) = \langle \mathbf{x},\mathbf{z}\rangle^d = \left(\sum_{i=1}^{n} x_i z_i\right)^d = \sum_{\mathbf{p}\in\mathbb{P}_d} \underbrace{\left(d! \prod_{p_i\in\mathbf{p}} \frac{1}{p_i!}\right)}_{q(\mathbf{p},d)} \mathbf{x}^{\mathbf{p}}\mathbf{z}^{\mathbf{p}} = \sum_{\mathbf{p}\in\mathbb{P}_d} q(\mathbf{p},d)\mathbf{x}^{\mathbf{p}}\mathbf{z}^{\mathbf{p}}, \tag{2}$$

with $\mathbb{P}_d \equiv \{\mathbf{p} \in \mathbb{N}_0^n \mid \|\mathbf{p}\|_1 = d\}$. Hence, Eq. 2 can be written as

$$\kappa^d_{HP}(\mathbf{x},\mathbf{z}) = \sum_{s=0}^{d} \sum_{\mathbf{p}\in\mathbb{P}^s_d} q(\mathbf{p},d)\mathbf{x}^{\mathbf{p}}\mathbf{z}^{\mathbf{p}}, \tag{3}$$

where $\mathbb{P}^s_d \equiv \{\mathbf{p} \in \mathbb{P}_d \mid \sum_{i=1}^{n} [\![p_i > 0]\!] = s\}$ and $[\![\cdot]\!]$ the indicator function.

Let us now partition the set $\mathbb{P}^s_d$ in such a way to have two vectors taken from $\mathbb{P}^s_d$ in the same class of equivalence if and only if they share the same components greater than zero. Specifically, given $\mathbf{b} \in \mathbb{B}_s$, then $\mathbb{P}^s_d(\mathbf{b}) \equiv \{\mathbf{p} \in \mathbb{P}^s_d \mid \forall i : p_i > 0 \iff b_i = 1\}$. With this notation, we can rewrite Eq. 3 as:

$$\kappa^d_{HP}(\mathbf{x},\mathbf{z}) = \sum_{s=0}^{d} \sum_{\mathbf{b}\in\mathbb{B}_s} \mathbf{x}^{\mathbf{b}}\mathbf{z}^{\mathbf{b}} \sum_{\mathbf{p}\in\mathbb{P}^s_d(\mathbf{b})} q(\mathbf{p},d). \tag{4}$$

Now, we can observe that, when s is fixed, then $\sum_{\mathbf{p}\in\mathbb{P}^s_d(\mathbf{b})} q(\mathbf{p},d)$ is constant over the elements $\mathbf{b} \in \mathbb{B}_s$. This is because the terms of the summations are the same. So, by taking any representative $\mathbf{b}_s \in \mathbb{B}_s$, we can rewrite Eq. 4 as:

$$\kappa^d_{HP}(\mathbf{x},\mathbf{z}) = \sum_{s=0}^{d} \underbrace{\left(\sum_{\mathbf{p}\in\mathbb{P}^s_d(\mathbf{b}_s)} q(\mathbf{p},d)\right)}_{h(s,d)} \left(\sum_{\mathbf{b}\in\mathbb{B}_s} \mathbf{x}^{\mathbf{b}}\mathbf{z}^{\mathbf{b}}\right) = \sum_{s=0}^{d} h(s,d)\ \kappa^s_{\wedge}(\mathbf{x},\mathbf{z}). \qquad \blacksquare$$

In the following we will show that, assuming boolean input vectors with the same number of active variables, a similar result of Theorem 1 also holds when using normalized monotone conjunctive kernels.

Theorem 2. *Given* $\mathbf{x}, \mathbf{z} \in \{0,1\}^n$ *such that* $\|\mathbf{x}\|_1 = \|\mathbf{z}\|_1 = m$ *, then any HP-kernel can be decomposed as a finite non-negative linear combination of normalized mC-kernels, that is:*

$$\kappa^d_{HP}(\mathbf{x}, \mathbf{z}) = \sum_{s=0}^{d} h(m, s, d)\, \widetilde{\kappa}^s_{\wedge}(\mathbf{x}, \mathbf{z}), \qquad h(m, s, d) \geq 0.$$

Proof. Consider the normalized mC-kernel, defined as follows:

$$\widetilde{\kappa}^s_{\wedge}(\mathbf{x}, \mathbf{z}) = \frac{\binom{\langle \mathbf{x}, \mathbf{z} \rangle}{s}}{\binom{\langle \mathbf{x}, \mathbf{x} \rangle}{s}^{\frac{1}{2}} \binom{\langle \mathbf{z}, \mathbf{z} \rangle}{s}^{\frac{1}{2}}}.$$

Since we assume $\|\mathbf{x}\|_1 = \|\mathbf{z}\|_1 = m$, we can write:

$$\widetilde{\kappa}^s_{\wedge}(\mathbf{x}, \mathbf{z}) = \frac{\binom{\langle \mathbf{x}, \mathbf{z} \rangle}{s}}{\binom{m}{s}^{\frac{1}{2}} \binom{m}{s}^{\frac{1}{2}}} = \frac{1}{\binom{m}{s}} \kappa^s_{\wedge}(\mathbf{x}, \mathbf{z})$$

where we used the fact that for binary vectors $\|\cdot\|_1 = \|\cdot\|_2^2$ always holds and hence, by Theorem 1 we can conclude:

$$\kappa^d_{HP}(\mathbf{x}, \mathbf{z}) = \sum_{s=0}^{d} \underbrace{h(s, d) \binom{m}{d}}_{h(m,s,d)} \widetilde{\kappa}^s_{\wedge}(\mathbf{x}, \mathbf{z}) = \sum_{s=0}^{d} h(m, s, d)\, \widetilde{\kappa}^s_{\wedge}(\mathbf{x}, \mathbf{z}).$$

■

As discussed by Donini et al. in [3], under mild conditions, any DPK of the form $\kappa(\mathbf{x}, \mathbf{z}) = f(\langle \mathbf{x}, \mathbf{z} \rangle)$ can be seen as a DPP, that is $\kappa(\mathbf{x}, \mathbf{z}) = \sum_{d=0}^{+\infty} a_d \kappa^d_{HP}(\mathbf{x}, \mathbf{z})$. Exploiting this result and the theorems above, we can get the following corollary.

Corollary 1. *Given* $\mathbf{x}, \mathbf{z} \in \{0,1\}^n$ *such that* $\|\mathbf{x}\|_1 = \|\mathbf{z}\|_1 = m$ *, then any DPK can be decomposed as a finite non-negative linear combination of normalized mC-kernels:*

$$\kappa(\mathbf{x}, \mathbf{z}) = f(\langle \mathbf{x}, \mathbf{z} \rangle) = \sum_{s=0}^{m} g(m, s)\, \widetilde{\kappa}^s_{\wedge}(\mathbf{x}, \mathbf{z}), \qquad g(m, s) \geq 0$$

Proof. (sketch) By using Theorem 1 we can see that $\kappa^d_{HP}(\mathbf{x}, \mathbf{z})$ can always be seen as a non-negative linear combination of the first m mC-kernels (since $\kappa^s_{\wedge}(\mathbf{x}, \mathbf{z}) = 0$ always holds when $s > m$). Hence, using the result in [3] and Theorem 2, the claim can be easily demonstrated.

■

4 The proposed algorithm

In the previous section we have shown that any DPK defined on binary vectors can be seen as a parametric linear combination of mC-kernels (a mCKP). In this section, we propose to make the combination non-parametric and to learn the coefficients of the mCKP by optimizing the radius-margin ratio of the combined kernel. Basically, we search on the kernel space $\kappa(\mathbf{x},\mathbf{z}) = \sum_s^d \mu_s \widetilde{\kappa}^s_{\wedge}(\mathbf{x},\mathbf{z})$, where $\mu_s \geq 0$, $\sum_s \mu_s = 1$ are the parameters to optimize. In this space we want to obtain the kernel that minimizes the radius-margin ratio.

First of all, we perform a change of variables by introducing a new vector of variables $\boldsymbol{\beta}$ and replacing $\mu_s(\boldsymbol{\beta}) = e^{\beta_s}/(\sum_r e^{\beta_r})$. This allows us to obtain an unconstrained problem easier to optimize. Specifically, we are now able to write the radius-margin ratio minimization problem as in the following:

$$\min_{\boldsymbol{\beta}} \Psi(\boldsymbol{\beta}), \text{ where } \Psi(\boldsymbol{\beta}) = \frac{1 - \hat{\boldsymbol{\alpha}}(\boldsymbol{\beta})^\top \left(\sum_{r=1}^R \mu_r(\boldsymbol{\beta})\mathbf{K}_r\right)\hat{\boldsymbol{\alpha}}(\boldsymbol{\beta})}{\hat{\boldsymbol{\gamma}}(\boldsymbol{\beta})^\top \mathbf{Y}\left(\sum_{r=1}^R \mu_r(\boldsymbol{\beta})\mathbf{K}_r\right)\mathbf{Y}\hat{\boldsymbol{\gamma}}(\boldsymbol{\beta})},$$

$$\hat{\boldsymbol{\alpha}}(\boldsymbol{\beta}) = \arg\min_{\boldsymbol{\alpha}\in\mathcal{A}} \boldsymbol{\alpha}^\top \left(\sum_{r=1}^R \mu_r(\boldsymbol{\beta})\mathbf{K}_r\right)\boldsymbol{\alpha} \text{ and } \hat{\boldsymbol{\gamma}}(\boldsymbol{\beta}) = \arg\min_{\boldsymbol{\gamma}\in\Gamma} \boldsymbol{\gamma}^\top \mathbf{Y}\left(\sum_{r=1}^R \mu_r(\boldsymbol{\beta})\mathbf{K}_r\right)\mathbf{Y}\boldsymbol{\gamma}.$$

By definition $\sum_{r=1}^R \mu_r(\boldsymbol{\beta}) = 1$, so

$$\Psi(\boldsymbol{\beta}) = \frac{\sum_{r=1}^R e^{\beta_r} \overbrace{\left(1 - \hat{\boldsymbol{\alpha}}(\boldsymbol{\beta})^\top \mathbf{K}_r \hat{\boldsymbol{\alpha}}(\boldsymbol{\beta})\right)}^{a_r(\boldsymbol{\beta})}}{\sum_{r=1}^R e^{\beta_r} \underbrace{\left(\hat{\boldsymbol{\gamma}}(\boldsymbol{\beta})^\top \mathbf{Y}\mathbf{K}_r\mathbf{Y}\hat{\boldsymbol{\gamma}}(\boldsymbol{\beta})\right)}_{b_r(\boldsymbol{\beta})}} \approx \frac{\langle e^{\boldsymbol{\beta}}, \mathbf{a}\rangle}{\langle e^{\boldsymbol{\beta}}, \mathbf{b}\rangle} = \bar{\Psi}(\boldsymbol{\beta}),$$

where $e^{\boldsymbol{\beta}} = [e^{\beta_1},\dots,e^{\beta_R}]$, $\mathbf{a} = [a_1,a_2,\dots,a_R]^\top$, $\mathbf{b} = [b_1,b_2,\dots,b_R]^\top$ and we assume $\mathbf{a},\mathbf{b}$ constants around a given $\boldsymbol{\beta}$. In order to optimize the function $\Psi(\boldsymbol{\beta})$ we then perform a series of steps of gradient descent on the approximated function $\bar{\Psi}(\boldsymbol{\beta})$ followed by a new computation of $\mathbf{a} = \mathbf{a}(\boldsymbol{\beta})$, and $\mathbf{b} = \mathbf{b}(\boldsymbol{\beta})$. The gradient step can be easily found as $\forall r \in \{1,\dots,R\}$ we have the following:

$$\frac{\partial \bar{\Psi}(\boldsymbol{\beta})}{\partial \beta_r} = \frac{a_r e^{\beta_r}\langle e^{\boldsymbol{\beta}},\mathbf{b}\rangle - b_r e^{\beta_r}\langle e^{\boldsymbol{\beta}},\mathbf{a}\rangle}{\langle e^{\boldsymbol{\beta}},\mathbf{b}\rangle^2} = \frac{e^{\beta_r}(a_r\langle e^{\boldsymbol{\beta}},\mathbf{b}\rangle - b_r\langle e^{\boldsymbol{\beta}},\mathbf{a}\rangle)}{\langle e^{\boldsymbol{\beta}},\mathbf{b}\rangle^2}$$

Summarizing, starting from $\boldsymbol{\beta} = \mathbf{0}$ and $\boldsymbol{\mu}(\boldsymbol{\beta})$ the uniform distribution over base kernels, at each iteration, the kernel combination $\mathbf{K}$ is computed using the current $\boldsymbol{\mu}(\boldsymbol{\beta})$ and then the vectors $\mathbf{a} = \mathbf{a}(\boldsymbol{\beta})$ and $\mathbf{b} = \mathbf{b}(\boldsymbol{\beta})$ can be computed as described above. Finally, the update of $\boldsymbol{\beta}$ and $\boldsymbol{\mu}(\boldsymbol{\beta})$ are performed as follows:

$$\beta_r \leftarrow \beta_r - \eta\frac{e^{\beta_r}\sum_s e^{\beta_s}(a_r b_s - a_s b_r)}{\langle e^{\boldsymbol{\beta}},\mathbf{b}\rangle^2} \quad \forall r \in \{1,\dots,R\}, \qquad \boldsymbol{\mu} \leftarrow \frac{1}{\sum_r e^{\beta_r}} e^{\boldsymbol{\beta}}$$

where η is the learning rate. This iterative procedure continues until a maximum number of iterations (*max_iter*) is reached.

5 Experimental assessment

Experiments have been performed using 11 binary and categorical datasets obtained from the UCI Machine Learning Repository [7]. The datasets have different sizes and characteristics, which are reported in Table 1. A preprocessing phase has been performed to make all these datasets binary. In particular, categorical features have been mapped into binary features by *one-hot* encoding, examples with missing values have been removed, and multiclass problems (audiology, zoo, primary-tumor, soybean, dna) transformed to binary ones by manually splitting the original classes in two groups. Different MKL settings for the combination of normalized mC-kernels have been compared. Namely:

- K_C^{avg}: the average of normalized mC-kernels of degrees 1 to 10, that is $\forall r, \mu_r = \frac{1}{10}$;
- K_C^{MKL}: the MKL solution where coefficients $\boldsymbol{\mu}$ are computed by the EasyMKL method [1] on normalized mC-kernels of degrees 1 to 10;
- $K_C^{RM\text{-}GD}$: the MKL solution of the gradient descent based algorithm proposed in this paper for the minimization of the radius-margin ratio when combining normalized mC-kernels of degrees 1 to 10;

For each MKL method, an SVM model has been trained using the obtained kernel. Available data have been split into training (50%) and test (50%); training data has been used to select the kernel and fit the SVM, then the AUC score has been calculated on the test set. To improve the statistical significance of the results, for each method, 50 runs with different splits (the same set for all the methods) have been performed. The average AUC in the test sets and the average ratio obtained in the training sets are reported in Table 1. Results show a significant AUC improvement of the proposed methodology $K_C^{RM\text{-}GD}$with respect to MKL baselines, for the large majority of tasks.

In order to better evaluate the behaviour of the proposed algorithm for the radius-margin optimization, in Figure 1 the distribution of weights is reported with respect to the one of EasyMKL, a MKL algorithm which aims at maximizing the margin alone. From the figure it is self-evident that margin maximization can give very different results with respect to the minimization of the radius-margin ratio. We observe that the weight vectors learned by our algorithm are very sparse, and hence only a small subset of kernels are combined to form the final kernel. The most typical configuration sets only two coefficients with large values, one low degree mC-kernel and one high degree mC-kernel.

Given the considerations above, we tried to apply the same gradient-descent radius-margin ratio optimization algorithm using only one non trivial mC-kernel combined with the identity matrix (note that high degree mC-kernels approximates the identity matrix). The optimal degree of the mC-kernel to combine is then chosen by selecting the one with the best ratio of the combined kernel. The obtained results, together with the average parameter selected in validation (the degree of the mC-kernel selected and the weight given to the identity matrix), are presented in the last column of Table 1. Not surprisingly, the obtained ratio is always worse than the ratio obtained by considering all the kernels at

dataset	K_C^{avg}	K_C^{MKL}	$K_C^{RM\text{-}GD}$	$K_{C,id}^{RM\text{-}GD}$
audiology	$99.99_{\pm 0.04}$	$99.99_{\pm 0.04}$	$100.00_{\pm 0.00}$	$100.00_{\pm 0.00}(2.64, 0.0023)$
(92,84,c)	$6.08_{\pm 0.33}$	$5.99_{\pm 0.32}$	$5.38_{\pm 0.25}$	$5.41_{\pm 0.25}$
zoo	$100.00_{\pm 0.00}$	$100.00_{\pm 0.00}$	$100.00_{\pm 0.00}$	$100.00_{\pm 0.00}(2.10, 0.0021)$
(101,21,c)	$3.01_{\pm 0.23}$	$2.62_{\pm 0.28}$	$2.24_{\pm 0.32}$	$2.27_{\pm 0.34}$
promoters	$96.37_{\pm 1.99}$	$96.38_{\pm 1.96}$	$95.83_{\pm 1.93}$	$95.82_{\pm 1.95}(1.00, 0.0068)$
(106,228,c)	$11.27_{\pm 0.25}$	$11.02_{\pm 0.32}$	$8.77_{\pm 0.63}$	$8.79_{\pm 0.65}$
primary-tumor	$72.55_{\pm 4.37}$	$72.69_{\pm 4.30}$	$74.58_{\pm 4.58}$	$75.53_{\pm 4.76}(1.38, 0.7079)$
(132,24,c)	$15.87_{\pm 1.30}$	$15.05_{\pm 0.87}$	$14.31_{\pm 0.72}$	$14.37_{\pm 0.69}$
house-votes	$99.11_{\pm 0.41}$	$99.10_{\pm 0.42}$	$99.20_{\pm 0.41}$	$99.21_{\pm 0.45}(2.30, 0.1992)$
(232,16,b)	$8.90_{\pm 1.13}$	$8.90_{\pm 1.17}$	$8.49_{\pm 1.13}$	$8.60_{\pm 1.12}$
soybean	$99.73_{\pm 0.19}$	$99.73_{\pm 0.19}$	$99.70_{\pm 0.25}$	$99.69_{\pm 0.25}(3.30, 0.0008)$
(266,88,c)	$11.49_{\pm 0.73}$	$11.32_{\pm 0.82}$	$10.86_{\pm 0.96}$	$10.93_{\pm 0.98}$
spect	$82.01_{\pm 3.14}$	$82.06_{\pm 3.02}$	$83.39_{\pm 3.10}$	$83.81_{\pm 3.11}(1.04, 0.5154)$
(267,23,b)	$18.91_{\pm 1.32}$	$18.56_{\pm 1.15}$	$17.53_{\pm 1.08}$	$17.63_{\pm 1.09}$
tic-tac-toe	$98.82_{\pm 0.46}$	$99.04_{\pm 0.39}$	$99.74_{\pm 0.20}$	$99.76_{\pm 0.20}(4.00, 0.0001)$
(958,27,c)	$73.39_{\pm 1.57}$	$70.93_{\pm 1.45}$	$60.75_{\pm 1.49}$	$60.92_{\pm 1.61}$
dna-bin	$98.46_{\pm 0.21}$	$98.53_{\pm 0.20}$	$98.71_{\pm 0.18}$	$98.69_{\pm 0.18}(2.00, 0.0001)$
(2000,180,b)	$118.49_{\pm 1.98}$	$108.87_{\pm 2.46}$	$103.45_{\pm 2.43}$	$104.16_{\pm 2.53}$
splice	$98.98_{\pm 0.13}$	$99.04_{\pm 0.12}$	$99.14_{\pm 0.15}$	$99.08_{\pm 0.13}(2.00, 0.0001)$
(3175,240,c)	$195.50_{\pm 2.13}$	$143.85_{\pm 2.99}$	$133.39_{\pm 3.26}$	$134.35_{\pm 3.30}$
kr-vs-kp	$99.90_{\pm 0.05}$	$99.91_{\pm 0.04}$	$99.92_{\pm 0.04}$	$99.95_{\pm 0.04}(3.80, 0.090)$
(3196,38,c)	$109.84_{\pm 2.75}$	$109.19_{\pm 2.89}$	$107.92_{\pm 2.95}$	$112.28_{\pm 2.79}$

Table 1. AUC score (1st row) and radius-margin ratio (2nd row) for all the methods. In the case of $K_{C,id}^{RM\text{-}GD}$the average parameters obtained in validation, i.e. average of degrees and average of the weights given to the identity matrix, are also indicated. For each dataset, in parenthesis, the information about #examples, #features and type of the dataset: binary ('b') or categorical ('c').

once. However, the difference is not so significant and the AUC score obtained by this simplified method is comparable, with the advantage of being highly parallelizable.

6 Conclusions

In this work we showed that, under mild conditions, any dot-product kernel applied to binary data can be decomposed in a linear non-negative parametric combination of monotone conjunctive kernels with different degrees. Then, a procedure to learn the (non-parametric) coefficients of the combination is proposed which exploits a radius-margin optimization algorithm based on gradient descent (here called RM-GD). The solutions returned by RM-GD are generally characterized by high sparseness and high AUC performance when compared to state-of-the-art margin-based MKL methods. Finally, our experiments also confirmed that the minimization of the radius-margin bound is an effective principle to pursue in order to minimize the expected test error.

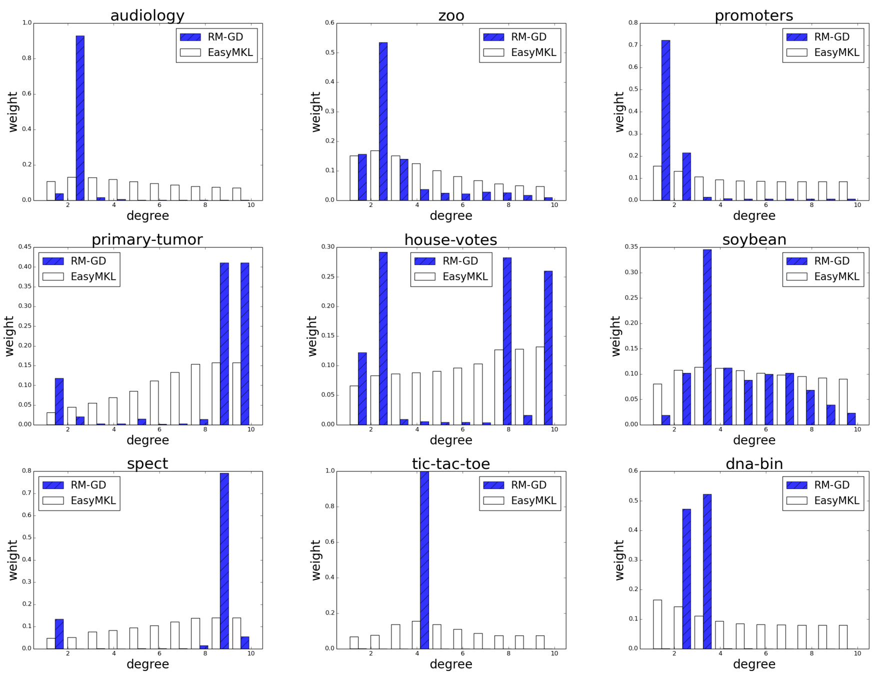

Fig. 1. Distribution of the weights obtained by EasyMKL (in white) and RM-GD (in blue) when combining mC-kernels of degrees 1 to 10 on nine UCI datasets.

References

1. Aiolli, F., Donini, M.: Easymkl: a scalable multiple kernel learning algorithm. Neurocomputing pp. 215–224 (2015)
2. Chung, K., Kao, W., Sun, C., Wang, L., Lin, C.: Radius margin bounds for support vector machines with the RBF kernel. Neural Computation 15(11), 2643–2681 (2003), `http://dx.doi.org/10.1162/089976603322385108`
3. Donini, M., Aiolli, F.: Learning deep kernels in the space of dot product polynomials. Machine Learning pp. 1–25 (2016)
4. Duan, K., Keerthi, S.S., Poo, A.N.: Evaluation of simple performance measures for tuning svm hyperparameters. Neurocomputing 51(Complete), 41–59 (2003)
5. Kalousis, A., Do, H.T.: Convex formulations of radius-margin based support vector machines. In: Proceedings of the 30th International Conference on Machine Learning. vol. 28, pp. 169–177 (2013)
6. Lauriola, I., Donini, M., Aiolli, F.: Learning dot-product polynomials for multiclass problems. In: Proceedings of the European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning (ESANN) (2017)
7. Lichman, M.: UCI machine learning repository (2013), `http://archive.ics.uci.edu/ml`
8. Schoenberg, I.J.: Positive definite functions on spheres. Duke Math. J. 9(1), 96–108 (1942)