HAL
open science

# Learning Landmark-Based Ensembles with Random Fourier Features and Gradient Boosting

Léo Gautheron, Pascal Germain, Amaury Habrard, Emilie Morvant, Marc Sebban, Valentina Zantedeschi

▶ **To cite this version:**

Léo Gautheron, Pascal Germain, Amaury Habrard, Emilie Morvant, Marc Sebban, et al.. Learning Landmark-Based Ensembles with Random Fourier Features and Gradient Boosting. 2019. hal-02148618

**HAL Id: hal-02148618**

**https://hal.science/hal-02148618**

Preprint submitted on 14 Jun 2019

**HAL** is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

# Learning Landmark-Based Ensembles with Random Fourier Features and Gradient Boosting

Léo Gautheron[1], Pascal Germain[2], Amaury Habrard[1], Emilie Morvant[1], Marc Sebban[1], and Valentina Zantedeschi

[1]Univ Lyon, UJM-Saint-Etienne, CNRS, Institut d Optique Graduate School, Laboratoire Hubert Curien UMR 5516, Saint-Etienne, France
[2]Equipe-projet Modal, Inria Lille - Nord Europe, Villeneuve d'Ascq, France

June 14, 2019

**Abstract**

We propose a Gradient Boosting algorithm for learning an ensemble of kernel functions adapted to the task at hand. Unlike state-of-the-art Multiple Kernel Learning techniques that make use of a pre-computed dictionary of kernel functions to select from, at each iteration we fit a kernel by approximating it as a weighted sum of Random Fourier Features (RFF) and by optimizing their barycenter. This allows us to obtain a more versatile method, easier to set-up and likely to have better performance. Our study builds on a recent result showing one can learn a kernel from RFF by computing the minimum of a PAC-Bayesian bound on the kernel alignment generalization loss, which is obtained efficiently from a closed-form solution. We conduct an experimental analysis to highlight the advantages of our method w.r.t. both Boosting-based and kernel-learning state-of-the-art methods.

## 1 Introduction

Kernel methods are among the most popular approaches in machine learning due to their capability to address non-linear problems, their robustness and their simplicity. However, they exhibit two main flaws in terms of memory usage and time complexity. To overcome the latter, some numerical approximation methods have been developed [23, 10]. Landmark-based approaches [4, 3, 5, 27] can be used to reduce the number of instances to consider in order to reduce the number of comparisons [21], but they heavily depend on the choice of the kernel. Tuning the kernel is, however, difficult and represents another drawback to tackle. Multiple Kernel Learning (MKL) [15, 13, 25, 24] and Matching Pursuit (MP) methods [17, 22] can provide alternatives to this problem but these require the use of a pre-defined dictionary of base functions.

Another strategy to improve the scalability of kernel methods is to use the Random Fourier Feature (RFF) approach that proposes to approximate some invariant-shift kernel with random features based on the Fourier Transform of the kernel [19]. This approach is data independent and then a predictor can be learned over these random features. Several works have extended this approach by allowing one to adapt the approximation with respect to the (learning) data points [26, 18, 20, 16, 1]. Among them, the recent work of [16] presents a method to quickly obtain a weighting distribution over the random features by a single pass over them. The method is derived from a statistical learning analysis, starting from the observation that each random feature can be interpreted as a weak hypothesis in the form of trigonometric functions obtained by the Fourier decomposition. Thus, a predictor can be

seen as a weighted majority vote over the random features. This Fourier decomposition is then considered as a *prior* distribution over the space of weak hypotheses/random features; the authors propose to learn a *posterior* distribution by optimizing a PAC-Bayesian bound with respect to a kernel alignment generalization loss over the learning data points. In other words, this corresponds to learning automatically a representation of the data through the approximation which then does not require to choose or tune a kernel in advance.

However, in practice, the method of [16] requires the use of a fixed set of landmarks selected beforehand and independently from the learning task. It is only once these landmarks are selected that the method can learn a representation based on the PAC-Bayesian bound. This leads to three important drawbacks: *(i)* the need for a heuristic strategy for selecting enough relevant landmarks, *(ii)* these landmarks and the associated representation might not be adapted for the task at hand, and *(iii)* the number of landmarks might not be minimal, inducing higher computational and memory costs. Instead of deliberately fixing the landmarks beforehand, we propose in this paper a Gradient Boosting approach (GB) [11] for learning both the landmarks and the associated random features combination directly, leading to a strong predictor. This strategy allows us to provide more compact and efficient representations in the context where the learning budget might be limited.

The reminder of the paper is organized as follows. Section 2 introduces the notations and the setting of the paper. Then, we recall in Section 3 the work of [16]. We introduce our landmark-based gradient boosting approach in Section 4. The experiments are performed in Section 5. Then we conclude in Section 6

## 2 Notations and Setting

We consider here binary classification tasks from a $d$-dimensional input space $\mathbb{R}^d$ to the label set $Y = \{-1, 1\}$. Let $S = \{z_i = (\mathbf{x}_i, y_i)\}_{i=1}^n \sim \mathcal{D}^n$ be a training set of $n$ points sampled *i.i.d.* from $\mathcal{D}$, a fixed and unknown data-generating distribution over $\mathbb{R}^d \times Y$.

In this paper, we focus on kernel-based algorithms that rely on pre-defined kernel functions $k : \mathbb{R}^d \times \mathbb{R}^d \to [-1, 1]$ assessing the similarity between any two points of the input space. These methods present good performances when the parameters of the kernels are learned and the chosen kernel can fit the distribution of the data. However, selecting the right kernel and tuning its parameters is computationally expensive. For this reason, Multiple Kernel Learning techniques [15, 13, 25, 24] have been proposed to select the combination of kernels that fits best the training data: a dictionary of base functions $\{k_t\}_{t=1}^T$ is composed by considering various kernels with their parameters fixed to several and different values and a combination is learned, taking the following form:

$$H(\mathbf{x}, \mathbf{x}') \;=\; \sum_{t=1}^{T} \alpha_t k_t(\mathbf{x}, \mathbf{x}'), \tag{1}$$

with $\alpha_t \in \mathbb{R}$ the weight of the kernel $k_t(\mathbf{x}, \mathbf{x}')$.

Similarly, in our method, we aim at learning linear combinations of kernels. However, we do not rely on a pre-computed dictionary of kernel functions. We rather learn them greedily, one per iteration of the Gradient Boosting procedure we propose (described in Section 4). Because of the computational advantages described in Section 1, we consider landmark-based shift-invariant kernels relying on the value $\delta = \mathbf{x}_t - \mathbf{x} \in R^d$ and denoted by abuse of notation:

$$k(\delta) = k(\mathbf{x}_t - \mathbf{x}) = k(\mathbf{x}_t, \mathbf{x}), \tag{2}$$

where $\mathbf{x}_t \in \mathbb{R}^d$ is the landmark of the input space which all the instances are compared to, that strongly characterizes the kernel. At each iteration of our Gradient Boosting procedure, we optimize not only this landmark but also the kernel function itself, exploiting the flexibility of the framework provided by [16]. We write the kernel as a sum of Random Fourier Features [19] and we learn a posterior distribution over them. We achieve this by

studying the generalization capabilities of the so-defined functions through the lens of the PAC-Bayesian theory. This theoretical analysis ultimately allows us to derive a closed-form solution of the posterior distribution $q_t$ (over the RFF at a given iteration $t$), which is guaranteed to minimize the kernel alignment loss. In the following section, we recall the framework of [16] and adapt it to our scenario.

# 3 Pseudo-Bayesian Kernel Learning with RFF

The kernel learning method proposed by [16] builds on the Random Fourier Features approximations proposed in [19]. Given a shift-invariant kernel $k(\delta) = k(\mathbf{x} - \mathbf{x}') = k(\mathbf{x}, \mathbf{x}')$, [19] show that

$$k(\mathbf{x} - \mathbf{x}') = \mathop{\mathbb{E}}_{\boldsymbol{\omega} \sim p} \cos\left(\boldsymbol{\omega} \cdot (\mathbf{x} - \mathbf{x}')\right),$$

with $p$ the Fourier transform of $k$ defined as

$$p(\boldsymbol{\omega}) = \frac{1}{(2\pi)^d} \int_{\mathbb{R}^d} k(\delta) \exp(-i\boldsymbol{\omega} \cdot \delta) d\delta.$$

This allows the kernel $k$ to be approximated in practice by drawing $K$ vectors from $p$ denoted by $\boldsymbol{\Omega} = \{\boldsymbol{\omega}_j\}_{j=1}^K \sim p^K$ and computing

$$k(\mathbf{x} - \mathbf{x}') \simeq \frac{1}{K} \sum_{j=1}^{K} \cos\left(\boldsymbol{\omega}_j \cdot (\mathbf{x} - \mathbf{x}')\right).$$

The larger $K$, the better the resulting approximation.

Instead of drawing RFF for approximating a known kernel, [16] propose to learn a new one by deriving a posterior distribution $q_t$ for a given landmark point in $\{\mathbf{x}_t\}_{t=1}^T$:

$$k_{q_t}(\mathbf{x}_t - \mathbf{x}) = \mathop{\mathbb{E}}_{\boldsymbol{\omega} \sim q_t} \cos\left(\boldsymbol{\omega} \cdot (\mathbf{x}_t - \mathbf{x})\right).$$

A distribution $q_t$ is learned by minimizing a PAC-Bayesian generalization bound on the expected value of the loss between the landmark $\mathbf{x}_t$ and any point $(\mathbf{x}, y) \sim \mathcal{D}$.

Let $(\mathbf{x}_t, y_t)$ be a sample, then its expected loss $\mathcal{L}^t$ and empirical loss $\widehat{\mathcal{L}}^t$ are respectively defined as

$$\mathcal{L}^t = \mathop{\mathbb{E}}_{(\mathbf{x},y)\sim\mathcal{D}} \ell(k_{q_t}(\mathbf{x}_t - \mathbf{x})), \quad \text{and} \quad \widehat{\mathcal{L}}^t = \frac{1}{n-1} \sum_{j=1, j\neq t}^{n} \ell(k_{q_t}(\mathbf{x}_t - \mathbf{x}_j)).$$

Using the PAC-Bayesian theory, they obtain the following theorem, under the linear loss $\ell(k_{q_t}(\mathbf{x}_t - \mathbf{x})) = \frac{1}{2} - \frac{1}{2} y_t y k_{q_t}(\mathbf{x}_t - \mathbf{x})$, by expressing the loss as

$$\begin{aligned} \mathcal{L}^t(k_{q_t}) &= \mathcal{L}^t\left(\mathop{\mathbb{E}}_{\boldsymbol{\omega}\sim p} h^t_{\boldsymbol{\omega}}\right) \\ &= \mathop{\mathbb{E}}_{\boldsymbol{\omega}\sim p} \mathcal{L}^t(h^t_{\boldsymbol{\omega}}), \end{aligned}$$

with $h^t_{\boldsymbol{\omega}}(\mathbf{x}) = \cos(\boldsymbol{\omega} \cdot (\mathbf{x}_t - \mathbf{x}))$. We note that the result also stands for any $[0,1]$-valued convex loss $\ell$. Indeed, by Jensen's inequality, we have $\mathcal{L}^t(k_{q_t}) = \mathcal{L}^t(\mathbb{E}_{\boldsymbol{\omega}\sim p}\, h^t_{\boldsymbol{\omega}}) \leq \mathbb{E}_{\boldsymbol{\omega}\sim p}\, \mathcal{L}^t(h^t_{\boldsymbol{\omega}})$.

**Theorem 1** (Theorem 1 from [16])**.** *For $s > 0$, $i \in \{1, \ldots, n\}$, a convex loss function $\ell : \mathbb{R} \times \mathbb{R} \to [0,1]$, and a prior distribution $p$ over $\mathbb{R}^d$, with probability $1 - \epsilon$ over the random choice of $S \sim \mathcal{D}^n$, we have for all $q$ on $\mathbb{R}^d$:*

$$\mathcal{L}^t(k_q) \leq \mathop{\mathbb{E}}_{\boldsymbol{\omega}\sim p} \widehat{\mathcal{L}}^t(h^t_{\boldsymbol{\omega}}) + \frac{1}{s}\left(KL(q\|p) + \frac{s^2}{2(n-1)} + \ln\frac{1}{\epsilon}\right),$$

*where $KL(q\|p) = \mathbb{E}_{\boldsymbol{\omega}\sim p} \frac{p(\boldsymbol{\omega})}{q(\boldsymbol{\omega})}$ is the Kullback-Leibler divergence between $q$ and $p$.*

**Algorithm 1:** Gradient Boosting with least square loss [11]

**Input :** Training set $S = \{\mathbf{x}_i, y_i\}_{i=1}^n$ with $y_i \in \{-1, 1\}$;
$T$: number of iterations; $v$: learning rate

**Output :** Weighted sum of predictors: $H(\mathbf{x}) = \text{sign}\left(H_0(\mathbf{x}) + \sum_{t=1}^{T} v\alpha_t h_{a_t}(\mathbf{x})\right)$

1: $H_0(\mathbf{x}) = \frac{1}{n}\sum_{i=1}^{n} y_i$
2: **for** $t = 1, \ldots, T$ **do**
3: $\quad \forall i = 1, \ldots, n, \quad \tilde{y}_i = y_i - H_{t-1}(\mathbf{x}_i)$
4: $\quad (\alpha_t, a_t) = \underset{\alpha, a}{\text{argmin}} \sum_{i=1}^{n} \left(\tilde{y}_i - \alpha h_a(\mathbf{x})\right)^2$, where $a$ denotes the parameters of the model $h_a$
5: $\quad H_t(\mathbf{x}) = H_{t-1}(\mathbf{x}) + v\alpha_t h_{a_t}(\mathbf{x})$
6: **end for**

It is well known [2, 8, 12] that the closed form solution minimizing the bound is the pseudo-posterior distribution $Q^t$ computed as

$$Q_j^t = \frac{1}{Z_t} \exp\left( - \beta\sqrt{n}\widehat{\mathcal{L}}^t(h_{\boldsymbol{\omega}}^t)\right), \tag{3}$$

for $j = 1, \ldots, K$ with $\beta$ a parameter and $Z_t$ the normalization constant. Finally, given a sample point $(\mathbf{x}_t, y_t)$ and $K$ vectors $\boldsymbol{\omega}$ denoted by $\boldsymbol{\Omega}^t = \{\boldsymbol{\omega}_j^t\}_{j=1}^K \sim p^K$, their kernel is finally defined as:

$$k_{Q^t}(\mathbf{x}_t - \mathbf{x}) = \sum_{j=1}^{K} Q_j^t \cos(\boldsymbol{\omega}_j \cdot (\mathbf{x}_t - \mathbf{x})).$$

Then [16] learn a representation of the input space of $n_L$ features where each new feature $t = 1, \ldots, n_L$ is computed using the kernel $k_{Q^t}$ with the sample $(\mathbf{x}_t, y_t)$. To do so, they consider a set of $n_L$ landmarks $L = \{(\mathbf{x}_t, y_t)\}_{t=1}^{n_L}$ which they chose either as a random subset of the training set, or as the centers of a clustering of the training set. Then, during a second step, a (linear) predictor can be learned from the new representation.

It is worth noticing that this kind of procedure exhibits two drawbacks. First, the model can be optimized only after having learned the representation. Second, the set of landmarks $L$ has to be fixed before learning the representation. Thus, the constructed representation is not guaranteed to be relevant for the learning algorithm considered. To tackle these issues, we propose in the next section a method performing the two steps at the same time through a gradient boosting algorithm, that allows us to learn the set of landmarks.

# 4 Gradient Boosting for Random Fourier Features

The approach we propose to follow the widely used gradient boosting framework first proposed by [11]. Before presenting our contribution, we quickly recall the classical gradient boosting algorithm instantiated with the least square loss.

## 4.1 Gradient Boosting in a Nutshell

Gradient boosting is an ensemble method that aims at learning a weighted majority vote over an ensemble of predictors in a greedy way by learning iteratively the predictors to add to the ensemble. The final majority vote is of the form

$$\forall \mathbf{x} \in \mathbb{R}^d,\ H(\mathbf{x}) = \text{sign}\left(H_0(\mathbf{x}) + \sum_{t=1}^{T} v\alpha_t h_{a_t}(\mathbf{x})\right),$$

where $H_0$ is a predictor fixed before the iterative process and is usually set such that it returns the same value for every data point, and $v\alpha_t$ is the weight associated to the predictor $h_{a_t}$ ($v$ is called the learning rate[1] and is fixed for each iteration, and $\alpha_t$ is called the optimal step size learned at the same time as the parameters $a_t$ of the predictor $h_{\alpha_t}$). Given a differentiable loss, the objective of the gradient boosting algorithm is to perform a gradient descent where the variable to be optimized is the ensemble and the function to be minimized is the empirical loss.

We now remind the gradient boosting algorithm instantiated with the least square loss in Algorithm 1 proposed by [11]. At the beginning (line **1**), the ensemble is constituted by only one predictor, the one that outputs the mean label over the whole training set. At each iteration, the first step (line **3**) consists in computing the negative gradient of the loss, also called the residual and denoted by $\tilde{y}_i$, for each training example $(\mathbf{x}_i, y_i) \in S$. Note that in the case of the least square loss the residual of an example is the deviation between its true label and the returned value of the current model. Then, it learns the parameters $a_t$ of the predictor $h_{\alpha_t}$, along with the optimal step size $\alpha_t$, that fit the best the residuals (line **4**). Finally, the current model is updated by adding $v\alpha_t h_{a_t}(\cdot)$ (line **5**) to the vote.

## 4.2 Our Algorithm

We now propose to benefit from the gradient boosting to tackle the drawbacks of the landmark-based approach of [16] recalled in Section 3. Our objective is to learn at the same time the landmarks (*i.e.*, the representation) and the classification model.

Let $k$ be a shift-invariant kernel and let $p$ be its Fourier transform. At each iteration $t$, given $\mathbf{\Omega}^t = \{\boldsymbol{\omega}_j^t\}_{j=1}^K \sim p^K$ a set of $K$ random features drawn from $p$, the objective is twofold:

- Learn the parameters $a_t$ of the base learner $h_{a_t}$ defined as

$$h_{a_t}(\mathbf{x}) = \sum_{j=1}^{K} Q_j^t \cos\Big(\boldsymbol{\omega}_j^t \cdot (\mathbf{x}_t - \mathbf{x})\Big). \tag{4}$$

  In our case, the parameters to be learned are $a_t = (\mathbf{x}_t, Q^t)$ where $\mathbf{x}_t$ is a landmark, and $Q^t$ is the pseudo-posterior distribution that can be computed using a closed-form similar to Equation (3).

- Compute the optimal step size $\alpha_t$.

In order to benefit from the theoretical guarantees of Theorem 1, and of the closed form of Equation (3), we propose the following greedy approach consisting in computing the landmark $\mathbf{x}_t$ by fixing the weight of each random features to $\frac{1}{K}$ (Equation (5)), then $Q^t$ thanks to its closed-form (Equation (7)), and finally $\alpha_t$ (Equation (8)).

First, given the set of random features $\mathbf{\Omega}^t$, we look for the landmark $\mathbf{x}_t \in \mathbb{R}^d$ that minimizes the average least square loss between the residuals and the kernel approximation defined as:

$$f_{\mathbf{\Omega}^t}(\mathbf{x}_t) = \frac{1}{n}\sum_{i=1}^{n}\left(\tilde{y}_i - \frac{1}{K}\sum_{j=1}^{K}\cos\Big(\boldsymbol{\omega}_j^t \cdot (\mathbf{x}_t - \mathbf{x}_i)\Big)\right)^2. \tag{5}$$

The minimization is done by performing a gradient descent of $f_{\mathbf{\Omega}^t}$ to find the landmark $\mathbf{x}_t$ that minimizes $f_{\mathbf{\Omega}^t}$ where the gradient of $f_{\mathbf{\Omega}^t}$ with respect to $\mathbf{x}_t$ is given by:

$$\frac{\partial f_{\mathbf{\Omega}^t}}{\partial \mathbf{x}_t} = \frac{2}{n}\sum_{i=1}^{n}\left(\frac{1}{K}\sum_{j=1}^{K}\boldsymbol{\omega}_j^t \sin\Big(\boldsymbol{\omega}_j^t \cdot (\mathbf{x}_t - \mathbf{x}_i)\Big)\right)\left(\tilde{y}_i - \frac{1}{K}\sum_{j=1}^{K}\cos\Big(\boldsymbol{\omega}_j^t \cdot (\mathbf{x}_t - \mathbf{x}_i)\Big)\right). \tag{6}$$

[1] The parameter $v$ is often referred as learning rate or shrinkage parameter. Decreasing $v$ usually improves the empirical performances [7] but requires to increase the number of boosting iterations $T$.

Second, given the landmark $\mathbf{x}_t$ found during the gradient descent, and given the set $\mathbf{\Omega}^t$, we compute the pseudo-posterior distribution $Q^t$ as:

$$\begin{aligned} Q_j^t &= \frac{1}{Z_t} \exp\Big( - c f_{\boldsymbol{\omega}_j^t}(\mathbf{x}_t)\Big) \\ &= \frac{1}{Z_t} \exp\left( \frac{c}{n} \sum_{i=1}^n \left( \tilde{y}_i - \cos\Big(\boldsymbol{\omega}_j^t \cdot (\mathbf{x}_t - \mathbf{x}_i)\Big)\right)^2 \right), \end{aligned} \tag{7}$$

for $j = 1, \ldots, K$ with $c \geq 0$ a parameter and $Z_t$ the normalization constant.

To finish, the optimal step size $\alpha_t$ is obtained by setting to 0 the derivative of line **4** with respect to $\alpha$. We then have

$$\alpha_t = \frac{\sum_{i=1}^n \tilde{y}_i h_{a_t}(\mathbf{x}_i)}{\sum_{i=1}^n h_{a_t}(\mathbf{x}_i)^2}. \tag{8}$$

This approach has two clear advantages compared to the two-step method of [16], where one learns the mapping first—for a pre-defined set of landmarks—and learns the predictor afterwards.

1. Gradient Boosting allows constructing iteratively the mapping by optimizing one landmark at each step.

2. The final predictor is learned at the same time and the learning procedure can be stopped when the empirical loss stops decreasing.

Consequently, the final mapping is likely to be less redundant and more suitable for the task at hand.

# 5 Experiments

In this section, we provide an empirical study of our method, referred as GBRFF. Firstly, we compare the performances of GBRFF with the two-step procedure from [16], referred as PBRFF, and also with boosting-based methods described in the next paragraph. Then, we compare the influence of the number of landmarks between GBRFF and PBRFF.

**Experimental Setup.** For GBRFF, we consider predictors as described in Equation (4) and select by cross-validation the parameter $c \in \{0\} \cup 2^{\{0,\ldots,10\}}$.

We compare GBRFF with the following algorithms :

- PBRFF [16] consists in first learning the new representation and then learning a linear SVM on the mapped training set. We select by cross-validation its parameters $\beta \in 10^{\{-3,\ldots,3\}}$ and $C \in 10^{\{-3,\ldots,3\}}$.

- XGB for Xgboost [9] and LGBM for LightGBM [14] which are state-of-the-art gradient boosting methods using trees as base predictors. For these methods, we select by cross-validation the maximum depth of the trees in $\{1, \ldots, 5\}$.

- MKBOOST [25] which is a Multiple Kernel Learning method based on the AdaBoost algorithm. At each boosting iteration, it selects the best performing kernel plugged inside a SVM, according to the Boosting weight distribution over the training examples. As it is done by the authors, we consider at each iteration 14 RBF kernels $k(\mathbf{x}, \mathbf{x}') = \exp(-\gamma\|\mathbf{x} - \mathbf{x}'\|^2)$ with $\gamma \in 2^{\{-6,\ldots,7\}}$ and 3 polynomial kernels $k(\mathbf{x}, \mathbf{x}') = (\mathbf{x}^T\mathbf{x}')^d$ with $d \in \{1, 2, 3\}$. We select by cross-validation the SVM parameter $C \in 10^{\{-5,\ldots,3\}}$.

- BMKR [24] which is another Multiple Kernel Learning method based on gradient boosting with least square loss. Similarly as in MKBOOST, it selects at each iteration the best performing kernel plugged inside an SVR to learn the residuals. It considers at each iteration 10 RBF kernels with $\gamma \in 2^{\{-4,\dots,5\}}$ and the linear kernel $k(\mathbf{x}, \mathbf{x}') = \mathbf{x}^T\mathbf{x}'$. We select by cross-validation the SVR parameter $C \in 10^{\{-5,\dots,3\}}$.

For the four methods based on gradient boosting, we further select by cross-validation the learning rate $v \in \{1, 0.5, 0.1, 0.05, 0.01\}$. The five boosting-based methods are run for $T = 200$ iterations. As for PBRFF which is not an iterative method, we select randomly with replacement $n_L = 200$ landmarks from the training set. For the two methods PBRFF and GBRFF using random features, we fix the number of random features $K$ to 100 and we draw them from the Fourier transform of the Gaussian kernel which is the normal law.

We consider 14 datasets coming mainly from the UCI repository. As we deal with binary classification problems, we have binarized the datasets as described in Table 1 where the classes considered respectively as the label '-1' and as the label '+1' are specified. All datasets

Table 1: Description of the datasets (n: number of examples, d: number of features, c: number of classes) and the classes chosen as negative (Label -1) and positive (Label +1)

| Name | n | d | c | Label -1 | Label +1 | Name | n | d | c | Label -1 | Label +1 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| wine | 178 | 13 | 3 | 2, 3 | 1 | wdbc | 569 | 30 | 2 | B | M |
| sonar | 208 | 60 | 2 | M | R | balance | 625 | 4 | 3 | B, R | L |
| glass | 214 | 11 | 6 | 2 3 5 6 7 | 1 | australian | 690 | 14 | 2 | 0 | 1 |
| newthyroid | 215 | 5 | 3 | 1 | 2, 3 | pima | 768 | 8 | 2 | 0 | 1 |
| heart | 270 | 13 | 2 | 1 | 2 | german | 1000 | 23 | 2 | 1 | 2 |
| bupa | 345 | 6 | 2 | 2 | 1 | splice | 3175 | 60 | 2 | +1 | -1 |
| iono | 351 | 34 | 2 | g | b | spambase | 4597 | 57 | 2 | 0 | 1 |

are normalized such that each feature has a mean of 0 and a variance of 1. For each dataset, we generate 20 random splits of 30% training examples and 70% testing examples. The hyper-parameters of all the methods are tuned by a 5-fold cross-validation on the training set. We report in Table 2 for each dataset the mean results over the 20 splits. In terms of accuracy, our method GBRFF shows competitive results with the state-of-the-art as it obtains the best performances on 5 datasets out of 14 with the best average rank among the six methods. This confirms the relevance of our algorithm.

Table 2: Mean test accuracy ± standard deviation over 20 random train/test splits.

| Dataset | XGB [9] | LGBM [14] | MKBOOST [25] | BMKR [24] | PBRFF [16] | GBRFF |
|---|---|---|---|---|---|---|
| wine | 94.92 ± 2.5 | 95.72 ± 2.1 | 98.56 ± 1.3 | **99.08** ± 0.6 | 97.92 ± 1.2 | 96.80 ± 2.2 |
| sonar | 76.34 ± 2.8 | 76.10 ± 3.2 | **77.77** ± 5.8 | 71.78 ± 4.3 | 75.82 ± 4.1 | 76.10 ± 7.5 |
| glass | **79.70** ± 3.2 | 78.33 ± 4.1 | 78.27 ± 2.8 | 77.93 ± 2.9 | 77.27 ± 3.4 | 75.70 ± 3.0 |
| newthyroid | 90.79 ± 2.9 | 83.01 ± 4.3 | 91.26 ± 13.8 | 94.17 ± 1.4 | **95.89** ± 1.6 | 93.18 ± 1.6 |
| heart | 79.71 ± 3.3 | 80.74 ± 2.4 | 77.67 ± 2.8 | **83.15** ± 2.0 | 83.02 ± 2.2 | 82.54 ± 2.2 |
| bupa | 66.10 ± 1.8 | 67.19 ± 2.7 | 58.39 ± 4.0 | 62.11 ± 3.3 | 65.48 ± 2.7 | **67.58** ± 3.2 |
| iono | 89.25 ± 1.7 | 88.64 ± 2.1 | 91.77 ± 5.9 | 92.40 ± 2.7 | **93.21** ± 1.9 | 85.55 ± 2.1 |
| wdbc | 94.60 ± 1.8 | 95.24 ± 1.8 | 95.16 ± 1.7 | 96.20 ± 0.8 | 95.99 ± 1.1 | **96.40** ± 1.1 |
| balance | 94.13 ± 2.4 | 95.02 ± 2.2 | 83.89 ± 9.3 | 93.36 ± 1.2 | **96.12** ± 1.4 | 94.77 ± 1.0 |
| australian | 85.33 ± 1.2 | 85.65 ± 1.4 | 80.46 ± 3.9 | 85.70 ± 1.1 | 85.66 ± 1.2 | **85.72** ± 1.3 |
| pima | 75.34 ± 1.8 | 74.81 ± 2.0 | 73.06 ± 2.5 | 75.02 ± 1.6 | 75.36 ± 2.1 | **75.66** ± 1.9 |
| german | 71.51 ± 1.2 | 71.60 ± 1.4 | 69.84 ± 1.3 | 71.18 ± 2.0 | 71.79 ± 1.3 | **72.36** ± 1.5 |
| splice | **96.35** ± 0.4 | 96.26 ± 0.4 | 82.70 ± 3.8 | 86.42 ± 0.6 | 85.27 ± 0.5 | 88.16 ± 0.5 |
| spambase | 94.20 ± 0.3 | **94.25** ± 0.3 | 90.45 ± 0.6 | 92.34 ± 0.5 | 91.60 ± 0.4 | 92.33 ± 0.3 |
| Average Rank | 3.40 | 3.20 | 4.40 | 3.07 | 2.80 | 2.73 |

**Influence of the number of landmarks.** In Figure 1, we analyze the accuracy of our landmark-based method GBRFF in two variants. The first one named GBRFF Learn corresponds to what was done in the previous experiment where at each iteration a landmark

was learned. The second named GBRFF Random considers at each iteration a landmark drawn randomly from the training set. In addition, we compare our method to PBRFF which also draws the landmarks randomly from the training set. To gain relevant insights, the analysis is made on three datasets for which our method has better and worse performances compared to PBRFF. We consider the datasets "sonar", "newthyroid" and "bupa".

Overall, as expected, the larger the quantity of landmarks, the better the performances for all methods. We see on the three datasets that GBRFF Learn presents better performances than GBRFF Random. The difference is especially large when the number of landmarks is small. For "sonar" and "bupa", PBRFF requires much more landmarks than GBRFF Learn to reach its maximal value. This shows the importance of learning the landmarks compared to selecting them randomly as it allows converging faster to possibly better performances. On the other hand, the results on "newthyroid" are better for PBRFF, no matter the number of landmarks used. This may happen because the linear classifier in the two methods is learned differently: it is learned using all landmarks by PBRFF with a Linear SVM and learned one landmark at a time by GBRFF with gradient boosting.

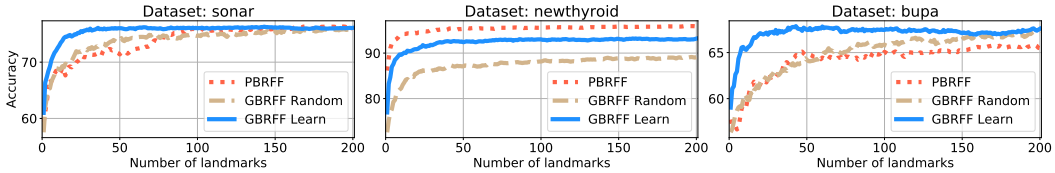


Figure 1: Mean test accuracy over 20 train/test splits on the "sonar", "newthyroid" and "bupa" datasets as a function of the number of landmarks used with PBRFF and our two variants of GBRFF.

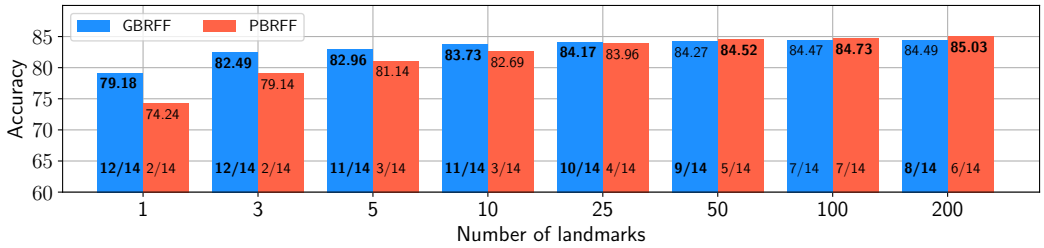


Figure 2: Mean test accuracy over 20 train/test splits and over the 14 datasets as a function of the number of landmarks used to train the two methods PBRFF and GBRFF. The mean values are displayed at the top of the bars, and the numbers of datasets where a method has the best performances are displayed at the bottom of the bars

We summarize in Figure 2 the influence of the number of landmarks used to train PBRFF and GBRFF. The figure gives the mean test accuracy across all datasets and over the 20 train/test splits. As seen in the previous experiment, with 200 landmarks, PBRFF and GBRFF have similar performances with respectively 84.49% and 85.03% of mean accuracy and with better performances for GBRFF on 8 datasets out of 14. However, when the number of landmarks decreases, GBRFF demonstrates a clear superiority. Indeed, we can observe in Figure 2 that with 25 landmarks, GBRFF provides a mean accuracy 0.21 point higher than the one of PBRFF while being better on 10 datasets, with 10 landmarks it is 1.04 points higher and better for 11 datasets, with 5 landmarks it gets 1.82 points higher and is still better for 11 datasets, with 3 landmarks it is 3.35 points higher and finally with only one landmark it is superior with 4.94 points higher. Additionally, our method obtains the best performances in 12 datasets out of 14 with less than 3 landmarks. Thus, the smaller the number of landmarks used, the better our method GBRFF compared to PBRFF. The gain

is significative when the number of landmarks is smaller than 25 which also corresponds to learning very small representations. This shows the clear advantage of our landmark-based method when learning compact representations with few landmarks, especially when one has a limited budget. In this case, learning the landmarks to solve the task at hand is preferable to selecting them randomly.

# 6 Conclusion

In this paper, we propose a Gradient Boosting algorithm where a kernel is learned at each iteration; the kernel being expressed with random Fourier features (RFF). Compared to state-of-the-art Multiple Kernel Learning techniques that select the best kernel function from a dictionary, and then plug it inside a kernel machine, we directly consider a kernel as a predictor that outputs a similarity to a point called landmark. We learn at each iteration a landmark by approximating the kernel as a sum of Random Fourier Features to fit the residuals of the gradient boosting procedure. Building on a recent work [16], we learn a pseudo-distribution over the RFF through a closed-form solution that minimizes a PAC-Bayes bound to induce a new kernel function tailored for the task at hand. The experimental study shows the competitiveness of the proposed method with state-of-the-art boosting and kernel learning methods, especially when the number of iterations used to train our model is small.

So far, the landmarks have been learned without any constraint. A promising future line of research is to add a regularization on the set of landmarks to foster diversity. In addition, the optimization of a landmark at each iteration can be computationally expensive when the number of iterations is large, and a possibility to speed-up the learning procedure is to derive other kernel approximations where the landmarks can be computed with a closed-form solution. Other possibilities regarding the scalability include the use of standard gradient boosting tricks [14, 9] such as sampling or learning the kernels in parallel. Another perspective could be to extend the analysis of [16] along with our algorithm to random Fourier features for operator-valued kernels [6] useful for multi-task learning or structured output.

**Acknowledgments**

This work was supported in part by the French Project APRIORI ANR-18-CE23-0015.

# References

[1] Raj Agrawal, Trevor Campbell, Jonathan Huggins, and Tamara Broderick. Data-dependent compression of random features for large-scale kernel approximation. In *22nd International Conference on Artificial Intelligence and Statistics*, 2019.

[2] Pierre Alquier, James Ridgway, and Nicolas Chopin. On the properties of variational approximations of gibbs posteriors. *The Journal of Machine Learning Research*, 17(1):8374–8414, 2016.

[3] Maria-Florina Balcan, Avrim Blum, and Nathan Srebro. Improved guarantees for learning via similarity functions. In *21st Annual Conference on Learning Theory - COLT*, pages 287–298, 2008.

[4] Maria-Florina Balcan, Avrim Blum, and Nathan Srebro. A theory of learning with similarity functions. *Machine Learning*, 72(1-2):89–112, 2008.

[5] Aurélien Bellet, Amaury Habrard, and Marc Sebban. Similarity learning for provably accurate sparse linear classification. In *29th International Coference on International Conference on Machine Learning*, pages 1491–1498, 2012.

[6] Romain Brault, Markus Heinonen, and Florence Buc. Random fourier features for operator-valued kernels. In *Asian Conference on Machine Learning*, pages 110–125, 2016.

[7] Peter Bühlmann and Torsten Hothorn. Boosting algorithms: Regularization, prediction and model fitting. *Statistical Science*, 22(4):477–505, 2007.

[8] Olivier Catoni. *PAC-Bayesian supervised classification: the thermodynamics of statistical learning*, volume 56. Inst. of Mathematical Statistic, 2007.

[9] Tianqi Chen and Carlos Guestrin. Xgboost: A scalable tree boosting system. In *22nd acm sigkdd international conference on knowledge discovery and data mining*, pages 785–794. ACM, 2016.

[10] Petros Drineas and Michael W. Mahoney. On the nyström method for approximating a gram matrix for improved kernel-based learning. *Journal of Machine Learning Research*, 6:2153–2175, 2005.

[11] Jerome H Friedman. Greedy function approximation: a gradient boosting machine. *Annals of statistics*, pages 1189–1232, 2001.

[12] Pascal Germain, Alexandre Lacasse, François Laviolette, and Mario Marchand. Pac-bayesian learning of linear classifiers. In *26th Annual International Conference on Machine Learning*, pages 353–360. ACM, 2009.

[13] Mehmet Gönen and Ethem Alpaydın. Multiple kernel learning algorithms. *Journal of machine learning research*, 12(Jul):2211–2268, 2011.

[14] Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, and Tie-Yan Liu. Lightgbm: A highly efficient gradient boosting decision tree. In *Advances in Neural Information Processing Systems*, pages 3146–3154, 2017.

[15] Gert RG Lanckriet, Nello Cristianini, Peter Bartlett, Laurent El Ghaoui, and Michael I Jordan. Learning the kernel matrix with semidefinite programming. *Journal of Machine learning research*, 5(Jan):27–72, 2004.

[16] Gaël Letarte, Emilie Morvant, and Pascal Germain. Pseudo-bayesian learning with kernel fourier transform as prior. In *22nd International Conference on Artificial Intelligence and Statistics*, volume 89, pages 768–776, 2019.

[17] Stéphane G Mallat and Zhifeng Zhang. Matching pursuits with time-frequency dictionaries. *IEEE Transactions on signal processing*, 41(12):3397–3415, 1993.

[18] Junier B Oliva, Avinava Dubey, Andrew G Wilson, Barnabás Póczos, Jeff Schneider, and Eric P Xing. Bayesian nonparametric kernel-learning. In *19th International Conference on Artificial Intelligence and Statistics*, 2016.

[19] Ali Rahimi and Benjamin Recht. Random features for large-scale kernel machines. In *Advances in neural information processing systems*, pages 1177–1184, 2008.

[20] Aman Sinha and John C Duchi. Learning kernels with random features. In *Advances In Neural Information Processing Systems*, pages 1298–1306, 2016.

[21] Ingo Steinwart. Sparseness of support vector machines. *Journal of Machine Learning Research*, 4(Nov):1071–1105, 2003.

[22] Pascal Vincent and Yoshua Bengio. Kernel matching pursuit. *Machine learning*, 48(1-3):165–187, 2002.

[23] Christopher KI Williams and Matthias Seeger. Using the nyström method to speed up kernel machines. In *Advances in neural information processing systems*, pages 682–688, 2001.

[24] Di Wu, Boyu Wang, Doina Precup, and Benoit Boulet. Boosting based multiple kernel learning and transfer regression for electricity load forecasting. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 39–51. Springer, 2017.

[25] Hao Xia and Steven CH Hoi. Mkboost: A framework of multiple kernel boosting. *IEEE Transactions on knowledge and data engineering*, 25(7):1574–1586, 2013.

[26] Zichao Yang, Andrew Gordon Wilson, Alexander J. Smola, and Le Song. A la carte - learning fast kernels. In *8th International Conference on Artificial Intelligence and Statistics*, 2015.

[27] Valentina Zantedeschi, Rémi Emonet, and Marc Sebban. Fast and provably effective multi-view classification with landmark-based svm. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 193–208. Springer, 2018.