Supplementary of Landmark-based Ensemble Learning with Random Fourier Features and Gradient Boosting

Léo Gautheron[1][⊠], Pascal Germain[2], Amaury Habrard[1], Guillaume Metzler[1], Emilie Morvant[1], Marc Sebban[1], and Valentina Zantedeschi[3]

[1] Univ Lyon, UJM-Saint-Etienne, CNRS, Institut d Optique Graduate School,
Laboratoire Hubert Curien UMR 5516, F-42023, Saint-Etienne, France
firstname.name@univ-st-etienne.fr leo_g_autheron@hotmail.fr
[2] Département d'informatique et de génie logiciel, Université Laval, Québec, Canada
pascal.germain@ift.ulaval.ca
[3] GE - Global Research, 1 Research Circle, Niskayuna, NY 12309
vzantedeschi@gmail.com

1 Additional experiments

Compared to the baseline method **PBRFF**, our two proposed methods **GBRFF1** and **GBRFF2** rely on different strategies in order to obtain an effective and efficient classifier. In the following, we propose different experiments that give some insights on the impact of the strategies used on both the classification accuracy and the computation time.

1.1 The importance of learning the landmarks in GBRFF1

Our method **GBRFF1** is based on **PBRFF** but is different in two points: (i) it learns the model at the same time as the representation instead of first learning the representation and then the model and (ii) it learns the landmarks used to build the representation instead of selecting them randomly from the training set. We compare these two methods to a variant called **GBRFF0.5** which is identical to **GBRFF1** except that we do not learn the landmarks in this variant, but we select them randomly as done in **PBRFF**. Figure 1 compares these three methods. We see that **GBRFF0.5** is faster than **PBRFF** but that it also has a lower accuracy. Thus, simply adapting the two step learning method of **PBRFF** in the one step learning method **GBRFF0.5** degrades the performances while slightly decreasing the computation time. These lower performances might come from the boosting classifier which is less effective than the SVM classifier in this setting. However, when comparing **GBRFF0.5** and **GBRFF1**, we see that learning the landmarks allows to improve the accuracy which becomes better than the one obtained by **PBRFF**, but at the expense of the computation time which becomes superior than both **GBRFF0.5** and **PBRFF**. These promising results in terms of accuracies motivate us to improve the learning strategy of the landmarks in **GBRFF1** to make it more efficient.

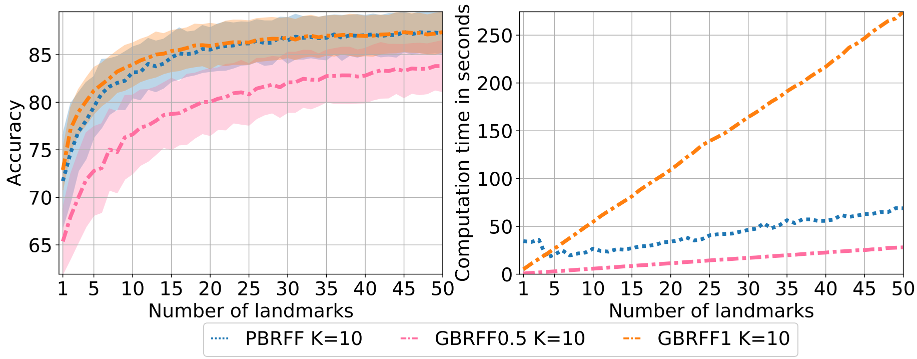

Fig. 1. Mean accuracy (left) and sum of computation time using the best parameters found with cross-validation (right) over 20 train/test splits and over the 15 first datasets (except "bankmarketing") for the three methods **PBRFF**, **GBRFF0.5** and **GBRFF1** using from 1 to 50 landmarks.

1.2 Improving the efficiency of GBRFF1

A key element of both **PBRFF** and **GBRFF1** is K, the amount of random features used for each landmark. We compare the performances and computation time of **GBRFF1** when using different numbers of random features per landmark. The results of this experiment are in Figure 2. As expected, the accuracy is better using more random features per landmark, but at the expense of presenting increasingly higher computation times. It seems that the higher the amount of

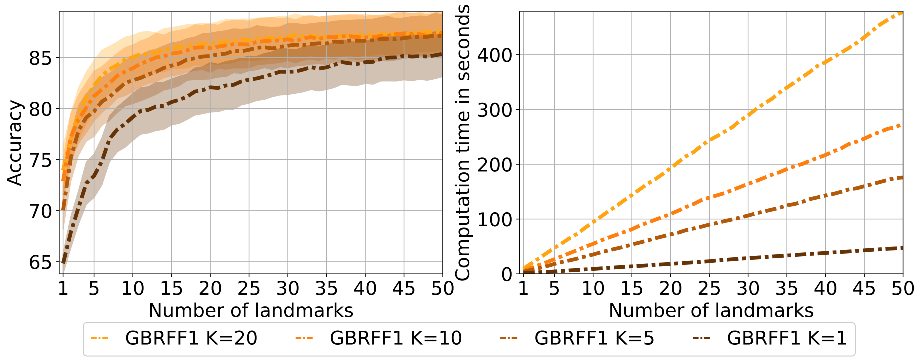

Fig. 2. Mean accuracy (left) and sum of computation time using the best parameters found with cross-validation (right) over 20 train/test splits and over the 15 first datasets (except "bankmarketing") for **GBRFF1** with $K \in \{1, 5, 10, 20\}$ random features used per landmark using from 1 to 50 landmarks.

random features is, the smaller the gain is in accuracy but the higher the addition to the computation time is. To illustrate this, we present in Figure 3 the accuracy divided by the computation time. The results show that **GBRFF1** with $K = 1$ presents the best compromise accuracy/computation time, meaning that even if for a fixed amount of landmarks T we can obtain better performances with a large value of K, it is more interesting to set $K = 1$ and use a large amount of landmarks T to obtain similar performances in less time. This behavior is confirmed by the results presented in Figure 4 showing that for different values of $T \times K$, we need to set $K = 1$ and use a large value of T to obtain the best accuracy. To understand why it is better to use a small amount K of random features per landmark, but a large amount of landmarks T, we remind the formula

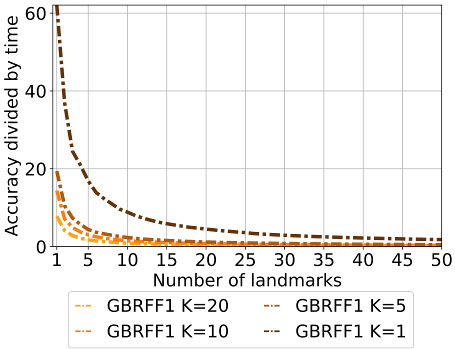

Fig. 3. Mean accuracy divided by sum of computation time using the best parameters found with cross-validation for **GBRFF1** with $K \in \{1, 5, 10, 20\}$ random feature used per landmark using from 1 to 50 landmarks over 20 train/test splits and over the 15 first datasets (except "bankmarketing").

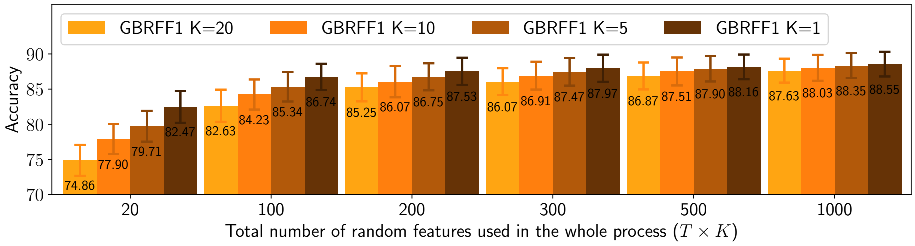

Fig. 4. Mean results over the 16 datasets *w.r.t.* the same total amount of random features $T \times K$ for $K \in \{1, 5, 10, 20\}$, with T the amount of boosting iterations.

of the final predictor of **GBRFF1** for a given example $\mathbf{x}$ which is

$$\text{sign}\left(H^0(\mathbf{x}) + \sum_{t=1}^{T}\alpha^t \sum_{j=1}^{K} q_j^t \cos\left(\boldsymbol{\omega}_j^t \cdot (\mathbf{x}^t - \mathbf{x})\right)\right)$$

but that simplifies when $K = 1$ to

$$\text{sign}\left(H^0(\mathbf{x}) + \sum_{t=1}^{T}\alpha^t \cos\left(\boldsymbol{\omega}^t \cdot (\mathbf{x}^t - \mathbf{x})\right)\right).$$

At a given iteration t, the objective is to learn the landmark $\mathbf{x}^t$, the boosting weight α^t and the random feature weights q^t that fit well the residuals defined by the exponential loss. Consequently, if K increases, so does the amount of constraints imposed at a given iteration to learn the landmark and the boosting weight. A possible explanation is that when $K = 1$, it is simpler to find a landmark and a weight α^t that correctly fit the residuals because both of them are less constrained, but that when using a large amount of random features, there is no possible solution that fits well the residuals under the constraints imposed by the random features.

The results presented motivate us to build upon **GBRFF1** using the smallest possible amount of random feature $K = 1$ per landmark.

1.3 From GBRFF1 to GBRFF2

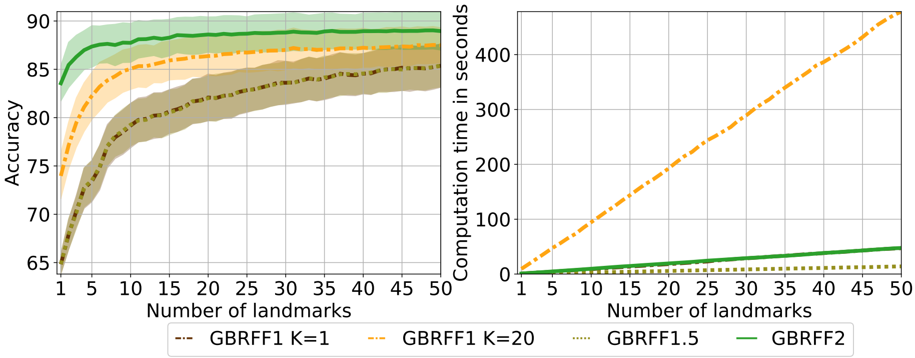

Fig. 5. Mean accuracy (left) and sum of computation time using the best parameters found with cross-validation (right) over 20 train/test splits and over the 15 first datasets (except "bankmarketing") for **GBRFF1**, **GBRFF1.5** and **GBRFF2** using from 1 to 50 landmarks.

Our proposed method **GBRFF2** is different from **GBRFF1** as (i) it uses a unique random feature per landmark and because (ii) the random part of the random feature ω is learned instead of fixed randomly. We introduce a variant called **GBRFF1.5** identical to **GBRFF2** except for ω which is not learned but remains fixed randomly. This variant is different from **GBRFF1** because the use of a unique random feature allows to learn a single scalar instead of a landmark vector to obtain the same model as **GBRFF1** with $K = 1$ more efficiently. The comparison in Figure 5 between **GBRFF1** with $K = 1$ and **GBRFF1.5** shows as expected that the two methods have exactly the same performances but with a much smaller computation time for **GBRFF1.5**. This confirms that when using a unique random feature, it is equivalent to learn a single scalar in $[-\pi, \pi]$ and a landmark vector in $\mathbb{R}^d$, but that it is much faster.

On the other hand, **GBRFF2** gives much better performances than **GBRFF1.5**, especially with a very small amount of landmarks, but at the expense of the computation time. Compared with **GBRFF1**, **GBRFF2** is faster for $K > 1$ or as fast for $K = 1$, and **GBRFF2** also achieves higher performances, even when using $K = 20$ random features for **GBRFF1**.