Scalable ridge Leverage score sampling for the Nyström method

Farah Cherfaoui, Hachem Kadri, Liva Ralaivola

Scalable ridge Leverage score sampling for the Nyström method

Farah Cherfaoui[1, 2], Hachem Kadri[2], and Liva Ralaivola[3]

[1]Aix Marseille Univ, CNRS, LIS, Marseille, France
[2]Aix Marseille Univ, CNRS, I2M, Marseille, France
[3]Criteo AI Lab, Paris, France

Abstract

The Nyström method, known as an efficient technique for approximating Gram matrices, builds upon a small subset of the data called landmarks, whose choice impacts the quality of the approximated Gram matrix. Various sampling methods have been proposed in the literature to choose such a subset, among which some based on ridge Leverage scores, which come with good theoretical and practical results. Nevertheless, direct computation of ridge leverage scores has an $\Theta(n^3)$ computation cost if n is the number of data, which is prohibitive when n is large. To tackle this problem, we here propose a $\Theta(n)$ divide-and-conquer (DAC) method to approximate ridge leverage scores and we provide theoretical guarantees and empirical results regarding their ability to blend with the Nyström approximation strategy. Our experimental results show that the proposed approximate leverage score sampling scheme achieves a good trade-off between predictive performance and running time.

Ridge leverage scores, Nyström approximation, kernel methods.

1 Introduction

Kernel methods [1, 2] exploit training data through the use of so-called *positive kernel* functions that compute dot products of a version of the training data that are embedded into a Reproducing kernel Hilbert space (RKHS). Their appeal comes from their strong theoretical basis, their ability to seamlessly generalize linear statistical approaches to nonlinear settings, and their effectiveness in handling structured data.

Given a dataset $\{x_i\}_{i=1}^n$ of n points, the working of a kernel method requires to compute the kernel k on all the pairs $\{(x_i, x_j)\}_{i=1}^n$ to construct the Gram matrix $K := [k(x_i, x_j)]_{i=1}^n$, of $\Theta(n^2)$ storage and $\Theta(n^3)$ computational

demand whenever K has to be inverted (a frequent need). Both storage and computational expectations are problematic when n is large. To overcome this problem, methods to approximate K have been devised, such as, to name a few, Random Fourier Features (RFF) [3], Cholesky factorization [4], and Nyström method [5], and proved effective [6, 7, 8, 9].

As stated before, we here focus on the Nyström approximation, which builds an approximation of the Gram matrix using *landmark* points. Results from the literature show that it is possible to reach approximation results better than those obtained by uniform sampling of the landmarks, by crafting data-dependent sampling techniques, which, for instance, could be based on *ridge leverage scores* and related quantities [10, 11]. However, these techniques remain expensive as computing the ridge leverage scores is $\Theta(n^3)$. Therefore, recent work has focused on methods able to efficiently approximate statistical quantities such as leverage scores [12, 13, 14, 15].

Here, we propose a new algorithm to efficiently approximate the ridge leverage scores, and using it in the Nyström method, we show that we achieve a good trade-off between predictive performance and running time. The paper is organized as follows. Section 2 recalls the Nyström method and the ridge leverage scores. Section 3 introduces our divide-and-conquer (DAC) ridge leverage score approximation algorithm and its theoretical guarantees. Finally, Section 4 reports empirical results of our method on real world datasets, showing that it allows for a solid approximation performance with competitive running times.

2 Nyström method and approximated ridge leverage scores

2.1 Nyström approximation

Let $\mathcal{X}$ be an input space, and $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ a kernel function associated to a reproducing kernel Hilbert space $\mathcal{H}_k$. Let $N := \{x_i\}_{i=1}^n$ be a set of n data point of $\mathcal{X}$ (i.e., $x_i \in \mathcal{X}$), and $K \in \mathbb{R}^{n \times n}$ the kernel matrix of N.

The Nyström method is a well studied technique for approximating the kernel matrix K and reducing the computational cost of kernel methods [5]. It builds an approximation $\hat{K}$ of K using only a small subset of points, called landmarks. Let $E \subseteq N$ be the landmark set. The approximated Gram matrix is set to $\hat{K} := K_{N,E}(K_E)^+ K_{N,E}^T$ where $K_{N,E} = (k(x_i, x_j))_{x_i \in N, x_j \in E}$ is a $n \times |E|$ matrix, and $K_E = (k(x_i, x_j))_{x_i \in E, x_j \in E}$ is a $|E| \times |E|$ matrix. Here, we denote by A^+ the Moore-Penrose inverse of matrix A and by A^T its transpose. This approximation is computed in $\Theta(n|E|^2)$, which is smaller than n^2 for $|E| \ll n$.

Data: $N := \{x_1, \ldots, x_n\}$ a set of n data, $\gamma \in]0, 1]$
Result: $\hat{K}$, an approximation to K
Compute $\hat{l}_1, \ldots, \hat{l}_n$ an over-approximation of ridge leverage scores ;
Let $p_i = \min\left\{1, 16\hat{l}_i \log\left(\frac{1}{\gamma}\sum_j \hat{l}_j\right)\right\}$;
Sample $E \subseteq N$ with probability p_i for each x_i ;
Return $\hat{K} := K_{N,E}(K_E)^+ K_{N,E}^T$ where $K_{N,E}$ and K_E are defined as in Section 2.1 ;

Algorithm 1: Kernel matrix approximation using approximated ridge leverage score.

2.2 Approximated ridge leverage scores

The ridge leverage scores are a measure of the 'importance' of each data point in the data set (see [16] for more details). If the ridge leverage scores are used to define the probability of sampling each data point, they offer a theoretically justified method for sampling landmarks giving rise to the matrix $\hat{K}$ with good approximation guarantees [10, 13].

For $\lambda > 0$, the ridge leverage score of any $x_i \in \mathcal{X}$ is:

$$l_i := [K\,(K + \lambda I)^{-1}]_{i,i} \ .$$

Unfortunately, since we need to invert the matrix $K + \lambda I$, the complexity of computing the ridge leverage scores is $\Theta(n^3)$. This makes the ridge leverage scores very expensive to compute and unusable for the Nyström method.

A number of recent papers have focused on approximating the ridge leverage scores (e.g., [12, 13]). In [12], a uniform leverage score approximation method was proposed. It is based on computing approximates of the ridge Leverage scores by approximating the kernel matrix K using a Nyström method with landmarks sampled uniformly at random. In [13], the authors propose a recursive version of the algorithm in [12]. They recursively improve the approximated ridge leverage scores by approximating the kernel matrix using landmarks sampled from the previously computed ridge leverage scores. They then used these approximations for Nyström method as described in Algorithm 1. They have shown that even if the complexity of the two methods is $\Theta(ns^2)$ (for some user-defined subsample size s), the uniform method is faster. Nevertheless, this comes at a cost in the approximation quality, since [13] has better approximation results than [12].

We present in the next section our DAC algorithm to approximate the ridge leverage scores with the same complexity as in [12, 13]. We then show in Section 4, that our method has good approximation results as [13], while being faster in practice.

Data: $N := \{x_1, \dots, x_n\}$ a set of n data, $r > 0$;
Result: $\hat{l}_i,\ \forall i \in \{1, \dots, n\}$;
Partition N: let $S_1, \dots, S_r$ be disjoint sets such that: $S_i \subseteq N$, $S_i \cap S_j = \emptyset$ for all $i, j \leq r$, and $\cup_{i=1}^{r} S_i = N$;
Compute $K_{S_1}(K_{S_1} + \lambda I)^{-1}, \dots, K_{S_r}(K_{S_r} + \lambda I)^{-1}$;
for $x_i \in N$;
do
  Let $S \in \{S_1, \dots, S_r\}$ be the set that contains x_i;
  Let j_i be the position of x_i in S ;
  $\hat{l}_i = [K_S(K_S + \lambda I)^{-1}]_{j_i, j_i}$ where
   $K_S = (k(x, y))_{x \in S, y \in S} \in \mathbb{R}^{|S| \times |S|}$;
end
Return $\hat{l}_i$ for all $i \in \{1, \dots, n\}$;

Algorithm 2: Divide And Conquer (DAC) for Ridge Leverage score approximation

3 Divide And Conquer (DAC) algorithm

We now describe a scalable simple approach to estimate the leverage scores. Our method is based on a divide-and-conquer (DAC) strategy. Instead of computing the leverage scores using all the data, we divide the data set into r disjoints small subsets of size s_i, $i = 1, \dots, r$, and we compute the leverage scores for each subset. The computational complexity will be then reduced from $\Theta(n^3)$ to $\Theta(\sum_i s_i^3)$. In the following, without loss of generality, we assume that all the subsets have the same size s. The overall complexity in this case is $\Theta(ns^2)$. Our DAC approach is depicted in Algorithm 2.

The DAC approach for approximating the leverage scores is easy and simple. One important question is what theoretical guarantees can we provide on the performance of the approximated leverage scores. To answer this question, we follow the work of [13] and provide the following theoretical results showing how the approximated leverage scores $\hat{l}_i$ relate to the true ones l_i.

Corollary 1. *Let $\mathcal{X}$ be an input space, and $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ a kernel function associated to a RKHS $\mathcal{H}_k$. let $N := \{x_1, \dots, x_n\}$ be a set of n data point in $\mathcal{X}$. For any $x_i \in N$, let $\hat{l}_i$ as computed in Algorithm 2. Then we have: $\hat{l}_i \geq l_i$.*

Proof. Let $S_1, \dots, S_r$ as in Algorithm 2, and let $\mathcal{S} := \{S_i\}_{i=1}^{r}$. For any $S \in \mathcal{S}$, let $\bar{S} := N \setminus S$ be the complement of S, and $K_S = (k(x, y))_{x \in S, y \in S}$ be the kernel matrix over S. It is known that $l_i = \sum_{j=1}^{n} \frac{\sigma_j}{\sigma_j + \lambda} U_{i,j}^2$, where $K = U \Sigma U^T$ is the SVD of K and $\sigma_i = \Sigma_{i,i}$ its singular values [17]. Using this, it is easy to see that $l_i = 1 - \lambda \left(K + \lambda I\right)^{-1}_{i,i}$. Let consider a fixed x_i,

and let S be the subset that contains x_i, and j_i be the position of x_i in S. Thus, $\hat{l}_i = 1 - \lambda \left(K_S + \lambda I\right)^{-1}_{j_i,j_i}$.

Without loss of generality, we assume that K is sorted such that the elements of S appear in the first block:

$$K = \begin{bmatrix} K_S & K_{S,\bar{S}} \\ K_{S,\bar{S}}^T & K_{\bar{S}} \end{bmatrix}$$

Where $K_{S,\bar{S}} = (k(x,z))_{S\times\bar{S}}$, and $K_{\bar{S}} = (k(x,z))_{\bar{S}\times\bar{S}}$. Using the Shur complement of a block matrix, we have:

$$(K + \lambda I)^{-1} = \begin{bmatrix} (K_S + \lambda I)^{-1} + CZC^T & * \\ * & * \end{bmatrix}$$

where $Z := \left(K_{\bar{S}} + \lambda I - K_{S,\bar{S}}^T \left(K_S + \lambda I\right)^{-1} K_{S,\bar{S}}\right)^{-1}$, and $C := \left(K_S + \lambda I\right)^{-1} K_{S,\bar{S}}$. Since $K + \lambda I$ is a PSD marix, so is its Shur complement Z. Using the fact that CZC^T is also a PSD matrix, and that the diagonal entries of a PSD matrix are positives, we obtain:

$$(K + \lambda I)^{-1}_{j_i,j_i} \geq \left(K_S + \lambda I\right)^{-1}_{j_i,j_i}.$$

We then have that: $\hat{l}_i \geq l_i$ for all i, since $\lambda > 0$.

□

Corollary 2. *Under the same conditions of Corollary 1, for any $x_i \in N$ let $\hat{l}_i$ as computed in Algorithm 2. We have*

$$\sum_{i=1}^{n} \hat{l}_i \leq \sum_{i=1}^{n} l_i + n\left(1 - \frac{1}{m}\right),$$

where $m := \frac{\sigma_{\max}+\lambda}{\sigma_{\min}+\lambda}$ is the condition number of $K + \lambda I$ and $\sigma_{\min}$ and $\sigma_{\max}$ are the smallest and largest eigenvalues of K.

Proof. Let $\mathcal{S} := \{S_i\}_{i=1}^{r}$ as defined in the proof of Corollary 1. For any $S \in \mathcal{S}$, let $\bar{S} := N \setminus S$. We thus have:

$$\sum_{i=1}^{n} \hat{l}_i = n - \lambda \sum_{S\in\mathcal{S}} \sum_{x_i \in S} \left(K_S + \lambda I\right)^{-1}_{j_i,j_i}, \tag{1}$$

where j_i is the position of x_i in the set S. First, we fix S and we bound $\sum_{x_i\in S}(K_S + \lambda I)^{-1}_{j_i,j_i}$. Without loss of generality, we assume that the matrix K is sorted such that the elements of S appear in the first block as in the proof of Corollary 1. Using the Shur complement [18], we also have:

$$(K + \lambda I)^{-1} = \begin{bmatrix} A & * \\ * & * \end{bmatrix},$$

where $A := \left(K_S + \lambda I - K_{S,\bar{S}}(K_{\bar{S}} + \lambda I)^{-1} K_{S,\bar{S}}^T\right)^{-1}$.

The main ingredient of the proof is the following result provided in Theorem 3.6.1 of [19]:

$$A \preceq \alpha_K (K_S + \lambda I)^{-1},$$

where $\alpha_K := \frac{(\sigma_{Min}+\sigma_{Max}+2\lambda)^2}{4(\sigma_{Min}+\lambda)(\sigma_{Max}+\lambda)}$ where σ_{Min} and σ_{Max} are the smallest and largest eigenvalues of the matrix K. Applying the trace operator we obtain

$$Trace\left((K_S + \lambda I)^{-1}\right) \geq \alpha_K^{-1} Trace\,(A)\,.$$

Using the fact that the trace of a matrix is equal to the sum of its diagonal elements, we have:

$$\sum_{x_i \in S} (K_S + \lambda I)^{-1}_{j_i,j_i} \geq \alpha_K^{-1} \sum_{x_i \in S} (K + \lambda I)^{-1}_{j_i,j_i}.$$

We now take the sum over all subsets $S \in \mathcal{S}$:

$$\sum_{S \in \mathcal{S}} \sum_{x_i \in S} (K_S + \lambda I)^{-1}_{j_i,j_i} \geq \alpha_K^{-1} \sum_{S \in \mathcal{S}} \sum_{x_i \in S} (K + \lambda I)^{-1}_{j_i,j_i}.$$

Using the fact that the subsets are disjoints and that their union is equal to N, and plugging this result in Equation (1), we obtain

$$\sum_{i=1}^{n} \hat{l}_i \leq n - \lambda \alpha_K^{-1} \sum_{i=1}^{n} (K + \lambda I)^{-1}_{i,i}.$$

Since $(K + \lambda I)^{-1}_{i,i} = \frac{1-l_i}{\lambda}$, we have:

$$\sum_{i=1}^{n} \hat{l}_i \leq \alpha_K^{-1} \sum_{i=1}^{n} l_i + n\left(1 - \alpha_K^{-1}\right).$$

Let $m := \frac{\sigma_{Max}+\lambda}{\sigma_{Min}+\lambda}$. It remains to prove that $\frac{1}{m} \leq \alpha_K^{-1} \leq 1$. Indeed,

$$\alpha_K^{-1} - 1 = \frac{-\left((\sigma_{Min} + \lambda) - (\sigma_{Max} + \lambda)\right)^2}{(\sigma_{Min} + \sigma_{Max} + 2\lambda)^2} \leq 0$$

and $\alpha_K^{-1} \geq \frac{4(\sigma_{Min}+\lambda)(\sigma_{Max}+\lambda)}{4(\sigma_{Max}+\lambda)^2} = \frac{\sigma_{Min}+\lambda}{\sigma_{Max}+\lambda} = \frac{1}{m}$. □

Using these bounds and a result in [13], we can obtain theoretical guarantees on the quality of approximated kernel matrix by Nyström approximation when using our DAC method.

Theorem 3. *Under the same assumptions of Corollary 1, let $\hat{K}$ be an approximation of K computed by Algorithm 1 with approximated ridge leverage score computed by Algorithm 2. Then, with probability at least $1-\gamma$, we have*

$$K - \lambda I \preceq \hat{K} \preceq K.$$

Moreover, $|E| \leq 32t\log(\frac{t}{\gamma})$, where $t := \sum_{i=1}^{n} l_i + n\left(1 - \frac{1}{m}\right)$.

Proof. Apply Theorem 3 of [13] using Corollaries 1 and 2. □

Table 1: Description of the datasets

Dataset	number of data	number of features
Covertype [21]	581012	54
Adult [22]	48842	15
KC1 [23]	2109	21
Houses [24]	20640	8

4 Experiments

In this section, we evaluate our proposed DAC algorithm on benchmark real world data sets described in Table 1. In all our experiments we use a Gaussian kernel with hyper-parameter σ and a regularization parameter λ fixed as in [20].

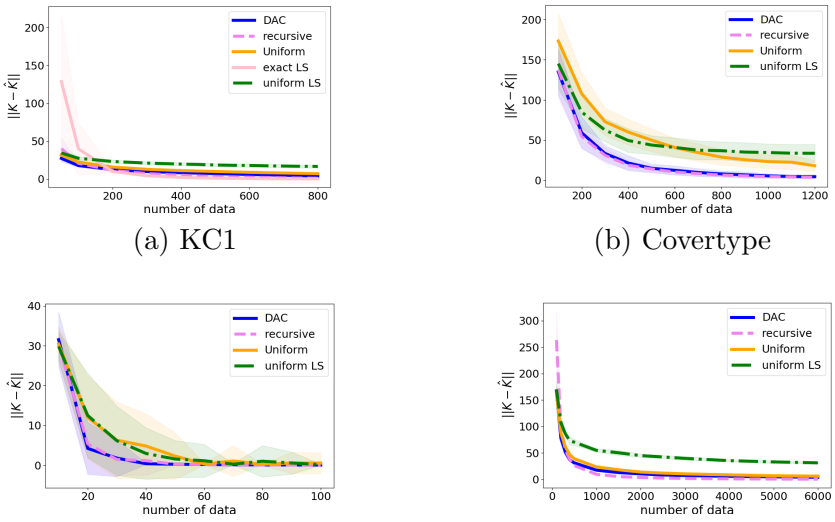
We use Algorithm 2 to estimate the leverage scores of each point, sample a subset of them proportionally to the approximated scores, and use the Nyström method to approximate the kernel matrix. We compare our DAC[1] approach to the uniform leverage score method (uniform LS) [12] and the recursive[2] method [13] since both run in $\Theta\left(ns^2\right)$. We also compare our method to the classical Nyström method, with landmarks sampled uniformly at random. Since the data sets are very large, we follow [13] and estimate this error on $10k$ random data point, except for KC1 which is relatively small for which we also can compute the exact leverage scores.

In our experiments we fix $s = \sqrt{n}$. We plot in Figure 1 the mean and variance of the Frobenius norm error over 20 runs in function of landmarks's size $|E|$ used in the Nyström method. Our results shows that our DAC approach outperforms the uniform sampling method and performs similar to the recursive method.

We compare in Figure 2 the running time of the different methods with different values of n. In [13], the authors also proposed an accelerated version of their recursive method. It improves the running time but not the approximation error. In this experiments, we also add the running time of the accelerated recursive method [13]. Our experimental results shows that the proposed DAC approximate leverage score sampling scheme achieves a good trade-off between predictive performance and running time.

[1]Our Python code is available online at `https://github.com/DAC-paper/Divide_And_Conquer_leverage_score_approximation`.

[2]We used the implementation available at `https://github.com/axelv/recursive-nystrom`.



(a) KC1 (b) Covertype

(c) Adult (d) Houses

Figure 1: Approximation error in function of the number of data points n with $s = \sqrt{n}$.

5 Conclusion

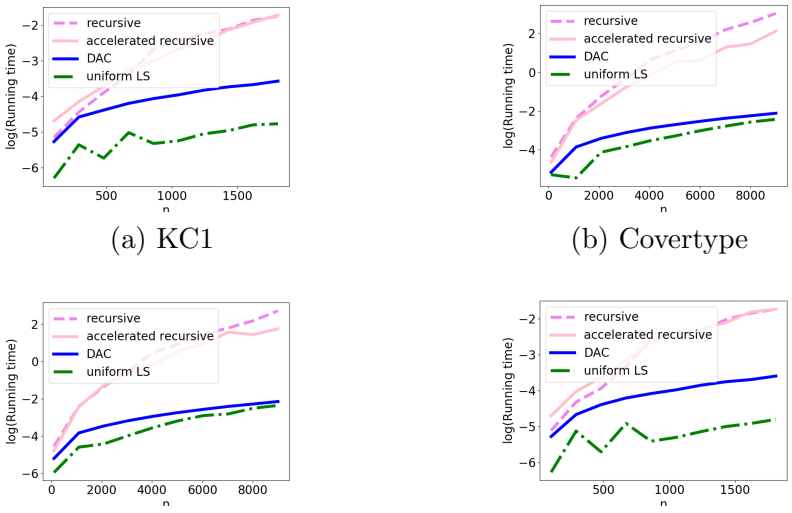
We proposed in this paper a new scalable and well-founded leverage score sampling scheme based on a divide and conquer strategy. Our experiments show that the proposed approach achieves good predictive performance with low running time. Further research may include the empirical study of the impact of the condition number of the kernel matrix on the approximation quality. It will be also interesting to study the usefulness of our approximate leverage score scheme in other applications such as active learning and dimensionality reduction.

References

[1] Thomas Hofmann, Bernhard Schölkopf, and Alexander J Smola, "Kernel methods in machine learning," *The annals of statistics*, vol. 36, no. 3, pp. 1171–1220, 2008.

[2] Jonathan H Manton and Pierre-Olivier Amblard, "A primer on reproducing kernel hilbert spaces," *Foundations and Trends® in Signal Processing*, vol. 8, no. 1–2, pp. 1–126, 2015.

[3] Ali Rahimi and Benjamin Recht, "Random features for large-scale kernel machines," *Advances in neural information processing systems (NeurIPS)*, vol. 20, 2007.



(a) KC1 (b) Covertype

(c) Adult (d) Houses

Figure 2: Running time in function of the number of data points n with $s = \sqrt{n}$.

[4] Shai Fine and Katya Scheinberg, "Efficient svm training using low-rank kernel representations," *Journal of Machine Learning Research*, vol. 2, pp. 243–264, 2001.

[5] Christopher Williams and Matthias Seeger, "Using the nyström method to speed up kernel machines," *Advances in neural information processing systems (NeurIPS)*, vol. 13, 2000.

[6] Nicholas Arcolano and Patrick J Wolfe, "Nyström approximation of Wishart matrices," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2010, pp. 3606–3609.

[7] Nicholas Arcolano and Patrick J Wolfe, "Estimating principal components of large covariance matrices using the nyström method," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2011, pp. 3784–3787.

[8] Guohong Liu, Hong Chen, Xiaoying Sun, and Robert C. Qiu, "Modified MUSIC algorithm for DOA estimation with nyström approximation," *IEEE Sensors Journal*, vol. 16, no. 12, pp. 4673–4674, 2016.

[9] Tao Zhang and Shiyuan Wang, "Nyström kernel algorithm under generalized maximum correntropy criterion," *IEEE Signal Processing Letters*, vol. 27, pp. 1535–1539, 2020.

[10] Alex Gittens and Michael W Mahoney, "Revisiting the nyström method for improved large-scale machine learning," *The Journal of Machine Learning Research (JMLR)*, vol. 17, no. 1, pp. 3977–4041, 2016.

[11] Chengtao Li, Stefanie Jegelka, and Suvrit Sra, "Fast DPP sampling for nyström with application to kernel methods," in *International Conference on Machine Learning (ICML)*, 2016, pp. 2061–2070.

[12] Ahmed Alaoui and Michael W Mahoney, "Fast randomized kernel ridge regression with statistical guarantees," *Advances in neural information processing systems (NeurIPS)*, vol. 28, 2015.

[13] Cameron Musco and Christopher Musco, "Recursive sampling for the nystrom method," *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 30, 2017.

[14] Alessandro Rudi, Daniele Calandriello, Luigi Carratino, and Lorenzo Rosasco, "On fast leverage score sampling and optimal learning," *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 31, 2018.

[15] Yifan Chen and Yun Yang, "Fast statistical leverage score approximation in kernel ridge regression," in *International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2021, pp. 2935–2943.

[16] Michael B Cohen, Yin Tat Lee, Cameron Musco, Christopher Musco, Richard Peng, and Aaron Sidford, "Uniform sampling for matrix approximation," in *Proceedings of the 2015 Conference on Innovations in Theoretical Computer Science*, 2015, pp. 181–190.

[17] Daniele Calandriello, Alessandro Lazaric, and Michal Valko, "Analysis of nyström method with sequential ridge leverage score sampling," in *International conference on Uncertainty in Artificial Intelligence (UAI)*, 2016.

[18] Kaare Brandt Petersen, Michael Syskind Pedersen, et al., "The matrix cookbook," *Technical University of Denmark*, vol. 7, no. 15, pp. 510, 2008.

[19] Rajendra Bhatia, *Positive definite matrices*, Princeton university press, 2009.

[20] Ulf Brefeld, Thomas Gärtner, Tobias Scheffer, and Stefan Wrobel, "Efficient co-regularised least squares regression," in *International conference on Machine learning (ICML)*, 2006, pp. 137–144.

[21] S. Bengio R. Collobert and Y. Bengio, "A parallel mixture of SVMs for very large scale problems," *Neural Computation, 14(05):1105-1114*, 2002.

[22] Ron Kohavi, "Scaling up the accuracy of naive-bayes classifiers: a decision-tree hybrid," *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*, 1996.

[23] J. Sayyad Shirabad and T.J. Menzies, "The promise repository of software engineering databases," 2005.

[24] Eman Ahmed and Mohamed Moustafa, "House price estimation from visual and textual features," *arXiv preprint arXiv:1609.08399*, 2016.