Quantifying Kernel Approximation Error of Random Fourier Features

Scott Kim
Department of Chemistry
University of Toronto
Toronto, ON M5S 3H6
scottyh.kim@mail.utoronto.ca

Richard Song
Department of Elec. & Comp. Engineering (ECE)
University of Toronto
Toronto, ON M5S 3G4
richardgm.song@mail.utoronto.ca

Karthik Panicker
Department of Chemistry
University of Toronto
Toronto, ON M5S 3H6
karthik.panicker@mail.utoronto.ca

1 Introduction

Kernel machines like Support Vector Machines (SVMs) are highly flexible and capable of approximating complex functions or decision boundaries when provided with sufficient training data. However, they face significant scalability challenges since methods involving the kernel matrix become computationally expensive for large datasets. For example, training a non-linear SVM on a dataset with half a million examples might take days, even on powerful hardware[1, 2, 3]. In contrast, linear models can be trained quickly on large datasets, especially when the data is low-dimensional. To leverage this advantage, some approaches factor the kernel matrix and use the resulting columns as features for linear models[4].

One of the most elementary methods would be the *Kernel Ridge Regression* (KRR). Given training data $\{(x_1, y_1), \dots, (x_n, y_n)\} \in \mathcal{X} \times \mathcal{Y}$, where $\mathcal{X} \subseteq \mathbb{R}^d$ is an input domain and $\mathcal{Y} \subseteq \mathbb{R}$ is an output domain, a positive definite kernel function $k : \mathcal{X} \times \mathcal{X} \longrightarrow \mathbb{R}$, and a regularization parameter $\lambda > 0$, the response for a given input $\mathbf{x}$ is estimated as [5]:

$$\bar{f}(\mathbf{x}) \equiv \sum_{j=1}^{n} k(\mathbf{x}_j, \mathbf{x})\alpha_j \tag{1}$$

where $\boldsymbol{\alpha} = (\alpha_1, \dots, \alpha_n)^\top$ is the solution to:

$$(\mathbf{K} + \lambda \mathbf{I}_n)\boldsymbol{\alpha} = \mathbf{y}. \tag{2}$$

In the above equation, $\mathbf{K} \in \mathbb{R}^{n \times n}$ is the *kernel matrix* defined by $\mathbf{K}_{ij} \equiv k(\mathbf{x}_i, \mathbf{x}_j)$ and $\mathbf{y} \equiv [y_1 \cdots y_n]^\top$ is the vector of responses. KRR is a straightforward yet robust technique, known for its strong statistical foundation and excellent practical performance. However, a significant drawback of KRR is its computational cost, which becomes prohibitive for large datasets. Specifically, solving the KRR problem typically requires $\Theta(n^3)$ computational time and $\Theta(n^2)$ memory. As a result, developing scalable approaches for KRR and other kernel-based methods has been a key area of active research in recent years[6, 7, 8].

One of the most popular approaches to effectively scale up kernel based methods is random Fourier features sampling, as introduced by Rahimi[9]. The method introduced in that paper uses a distribution D on functions from $\mathcal{X}$ to $\mathbb{C}^s$ (s is a parameter) such that for every $\mathbf{x}, \mathbf{z} \in \mathbb{R}^d$

$$k(\mathbf{x}, \mathbf{z}) = \mathbb{E}_{\phi \sim D}\big[\phi(\mathbf{x})^* \phi(\mathbf{z})\big]. \tag{3}$$

The random features method involves sampling a function ϕ from the distribution D and using the surrogate kernel $\tilde{k}(\mathbf{x}, \mathbf{z}) \equiv \phi(\mathbf{x})^* \phi(\mathbf{z})$. This approach is based on Bochner's theorem, which states that any bounded, continuous, and shift-invariant kernel can be expressed as the Fourier transform of a bounded positive measure, known as the spectral measure[10, 11]. The feature map is generated by sampling this spectral measure. This approach allows the approximate KRR estimator to be computed in $O(ns^2)$ time and $O(ns)$ memory, offering significant computational advantages when $s \ll n$ [9].

To this end, the experimental dependence of training time on *training dataset size* and *number of features* is an interesting relationship to investigate. The most recent example in this area would be the work by Avron et al. [5], which compares RFF to a modified sampling method with varying feature sizes. Additionally, work done by Rahimi [9] compares the speed-up between RFF relative to SVM (kernel-based method) on varying datasets as well. In this paper, we investigate the impact of RFF vs. KRR on the QM9 dataset[12, 13] with respect to training and feature size.

In the rest of this paper, the theoretical underpinnings of RFF are examined in greater detail in Section 2, while the experiments and results are elaborated on in Section 3 and Section 4, respectively.

2 Background

2.1 Kernel Ridge Regression

Consider a learning problem with data $\{(\mathbf{x}_1, y_1), ..., (\mathbf{x}_N, y_N)\} \in \mathcal{X} \times \mathcal{Y}$, where $\mathcal{X} \subseteq \mathbb{R}^d$ and $\mathcal{Y} \subseteq \mathbb{R}$. In matrix notation, the cost function to minimize in ℓ_2 regularized least squares regression is

$$\mathcal{J}(\mathbf{w}) = ||\mathbf{y} - \mathbf{X}\mathbf{w}||_2^2 + \lambda ||\mathbf{w}||_2^2 \tag{4}$$

where $\mathbf{y} \in \mathbb{R}^N$ is the vector of targets, $\mathbf{X} \in \mathbb{R}^{N \times d}$ is the design matrix, and $\mathbf{w} \in \mathbb{R}^d$ is the vector of regression coefficients. Let $\mathbf{w} = \mathbf{X}^\top \boldsymbol{\alpha}$, which gives a cost function defined in terms of $\boldsymbol{\alpha}$:

$$\mathcal{J}(\boldsymbol{\alpha}) = ||\mathbf{y} - \mathbf{X}\mathbf{X}^\top \boldsymbol{\alpha}||_2^2 + \lambda ||\mathbf{X}^\top \boldsymbol{\alpha}||_2^2 \tag{5}$$

$\mathbf{X}\mathbf{X}^\top$ is a symmetric and positive semi-definite matrix for any collection $\{\mathbf{x}_1, ..., \mathbf{x}_N\}$. This matrix can be generated by the function $k(\mathbf{x}, \mathbf{y}) = \mathbf{x}^\top \mathbf{y}$, which defines an inner product in our feature space. Let $\mathbf{X}\mathbf{X}^\top = \mathbf{K}$, where $\mathbf{K}_{ij} = k(\mathbf{x}_i, \mathbf{x}_j)$. Using the identity $||\mathbf{X}^\top \boldsymbol{\alpha}||_2^2 = \boldsymbol{\alpha}^\top \mathbf{K} \boldsymbol{\alpha}$, the cost function becomes

$$\mathcal{J}(\boldsymbol{\alpha}) = ||\mathbf{y} - \mathbf{K}\boldsymbol{\alpha}||_2^2 + \lambda \boldsymbol{\alpha}^\top \mathbf{K} \boldsymbol{\alpha} \tag{6}$$

Setting the gradient of the cost function with respect to $\boldsymbol{\alpha}$ to 0 gives us the closed-form solution for $\boldsymbol{\alpha}$:

$$\hat{\boldsymbol{\alpha}} = (\mathbf{K} + \lambda \mathbf{I})^{-1} \mathbf{y} \tag{7}$$

Predictions for the test set are calculated in the following way:

$$\begin{aligned} \bar{\mathbf{y}} &= \mathbf{X}_{\text{test}} \mathbf{w} \\ &= \mathbf{X}_{\text{test}} \mathbf{X}^\top \hat{\boldsymbol{\alpha}} \\ &= \mathbf{K}_{\text{test}} (\mathbf{K} + \lambda \mathbf{I})^{-1} \mathbf{y} \end{aligned} \tag{8}$$

By Mercer's theorem [14], any symmetric and positive semi-definite function $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ defines an inner product in the vector space of its arguments. If we expand our original features with some basis function $\psi : \mathcal{X} \to \mathcal{P}$, where $\mathcal{P} \subseteq \mathbb{R}^p$, $\mathbf{K}$ would still be symmetric and positive semi-definite, thus defining an inner product in the new feature space. $\mathbf{K}_{ij}$ can then be obtained by computing $k_\psi(\mathbf{x}_i, \mathbf{x}_j)$. As such, KRR is a learning algorithm that searches for the optimal linear boundary in non-linear and often high-dimensional spaces as specified by the choice of kernel function while avoiding the cost of expensive feature expansions. However, equation 8 shows that making predictions with KRR requires inversion of an $N \times N$ matrix and $M \times N$ evaluations of the kernel function between the test and training set.

2.2 Random Fourier Features (RFFs)

The Gaussian kernel is the kernel chosen for this work. The form of the Gaussian kernel is

$$k(\mathbf{x}, \mathbf{y}) = \exp\left(-\frac{\|\mathbf{x} - \mathbf{y}\|_2^2}{2\sigma^2}\right) \tag{9}$$

The Gaussian kernel is shift-invariant, which by Bochner's theorem [10] means that it is the Fourier transform of a non-negative measure. For the Gaussian kernel, the corresponding measure $p(\boldsymbol{\omega})$ is the standard Gaussian:

$$p(\boldsymbol{\omega}) = \mathcal{N}(\mathbf{0}, \mathbf{I}) \tag{10}$$

Starting from Bochner's theorem, the Gaussian kernel can be approximated with a Monte Carlo estimate using R samples:

$$\begin{aligned} k(\mathbf{x}, \mathbf{y}) &= \int_{\mathbb{R}^d} p(\boldsymbol{\omega}) e^{i\boldsymbol{\omega}^\top (\mathbf{x}-\mathbf{y})} d\boldsymbol{\omega} \\ &\approx \frac{1}{R} \sum_{r=1}^{R} e^{i\boldsymbol{\omega}_r^\top (\mathbf{x}-\mathbf{y})} \end{aligned} \tag{11}$$

Note that the kernel is real-valued, so we can rewrite it using Euler's formula:

$$k(\mathbf{x}, \mathbf{y}) \approx \frac{1}{R} \sum_{r=1}^{R} \cos(\boldsymbol{\omega}_r^\top (\mathbf{x} - \mathbf{y})) \tag{12}$$

which is equivalent to

$$k(\mathbf{x}, \mathbf{y}) \approx \frac{1}{R} \sum_{r=1}^{R} 2\cos(\boldsymbol{\omega}_r^\top \mathbf{x} + b_r)\cos(\boldsymbol{\omega}_r^\top \mathbf{y} + b_r) \tag{13}$$

where $b_r \in \mathbb{R}$ is drawn from Uniform$(0, 2\pi)$. Defining $\phi_r(\mathbf{x}) = \sqrt{\frac{2}{R}}\cos(\boldsymbol{\omega}_r^\top \mathbf{x} + b_r)$, we obtain

$$\begin{aligned} k(\mathbf{x}, \mathbf{y}) &\approx \sum_{r=1}^{R} \phi_r(\mathbf{x})\phi_r(\mathbf{y}) \\ &= \boldsymbol{\phi}(\mathbf{x})^\top \boldsymbol{\phi}(\mathbf{y}) \end{aligned} \tag{14}$$

Having shown that we can approximate the inner product space defined by the kernel function to arbitrary accuracy, the cost function can be formulated with the feature expansion $\boldsymbol{\phi}(\mathbf{x})$. Let $\boldsymbol{\Phi} \in \mathbb{R}^{N \times R}$ be the feature matrix after applying the feature expansion to $\mathbf{X}$. Then the cost function to minimize is

$$\mathcal{J}(\boldsymbol{\beta}) = ||\mathbf{y} - \boldsymbol{\Phi}\boldsymbol{\beta}||_2^2 + \lambda||\boldsymbol{\beta}||_2^2 \tag{15}$$

where $\boldsymbol{\beta}$ is the vector of regression coefficients in the transformed RFF space. The solution is similar to the regularized least-squares solution:

$$\hat{\boldsymbol{\beta}} = (\boldsymbol{\Phi}^\top \boldsymbol{\Phi} + \lambda \mathbf{I})^{-1} \boldsymbol{\Phi}^\top \mathbf{y} \tag{16}$$

And predictions are made by

$$\bar{\mathbf{y}} = \boldsymbol{\Phi}_{\text{test}} \boldsymbol{\beta} \tag{17}$$

3 Methods

3.1 Datasets

The two chosen learning problems are classification on a small dataset and regression on a large dataset. The small dataset is the MAGIC Gamma Telescope dataset [15], which contains simulated registration of high energy gamma particles in a ground-based atmospheric Cherenkov gamma telescope. It is split into signal and background noise classes whose images are processed to obtain various features.

The QM9 dataset [16] is a quantum chemical dataset consisting of 134k small organic molecules with up to nine heavy atoms (C, N, O, and F). QM9 is a standard benchmark for evaluating machine learning models in supervised learning of quantum mechanical properties. We chose to learn the atomization energy from the coordinates of the nuclei.

3.2 Hyperparameter optimization

In KRR, the hyperparameters to be optimized are the strength of the regularizer λ and the length parameter of the Gaussian kernel σ^2. In all experiments, a grid search was performed to identify the hyperparameter pair that gave the lowest classification accuracy or mean absolute error. For all experiments with RFFs, 4-fold cross-validation was conducted on each training set. For experiments in the MAGIC dataset with regular features, coarser grids were used. For experiments in the QM9 datset with regular features, 4-fold cross-validation was performed once on a small training set size due to time constraints. Appendix has heatmaps which was used to optimize hyperparameters for RFF-KRR model on QM9 dataset.

3.3 Experiments

3.3.1 RFF Accuracy Across Feature Size

In this experiment, models were trained with 80% of the dataset using different numbers of RFFs to evaluate the relationship between feature complexity and model performance.

3.3.2 RFF Accuracy Across Training Set Sizes

In supervised learning, a well-known empirical law is the linear relationship between the error on the test set and the training set size on a log-log scale [17]. We investigate the effect of different numbers of RFFs on the slopes and intercepts of the learning curves and compare performance to regular features.

3.3.3 Time Comparison

The main merit of using RFFs is the linear time complexity during the kernel inversion in KRR [9]. The difference in training and prediction time between regular KRR and KRR with different numbers of RFFs were compared at various training set sizes.

4 Results

4.1 RFF Accuracy Across Feature Size

Figure 1 shows a learning curve for the MAGIC dataset at varying numbers of RFFs. The error decreases logarithmically as number of features increases, and the variance of the errors decreases as well. The decrease in the variance can be rationalized by the fact that the Gaussian distribution of distances is more completely captured in the feature vector at higher feature sizes. Lower feature sizes are more sensitive to random draws from this distribution and thereby show higher variance in the error.

Figure 2 shows a learning curve for QM9 with respect to the feature size. The relationship between the error and the number of features is approximately linear on a log-log scale, and the variance vanishes at 10 RFFs. It is not clear why the response of the error to the number of RFFs is different between the classification and regression tasks, and this relationship should be studied further.

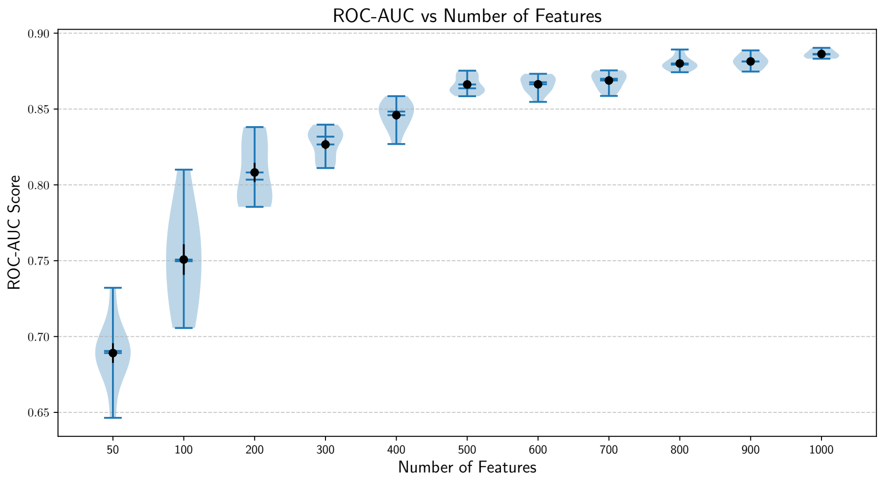

Figure 1: Accuracy score as a function of feature size at training size of 0.8.

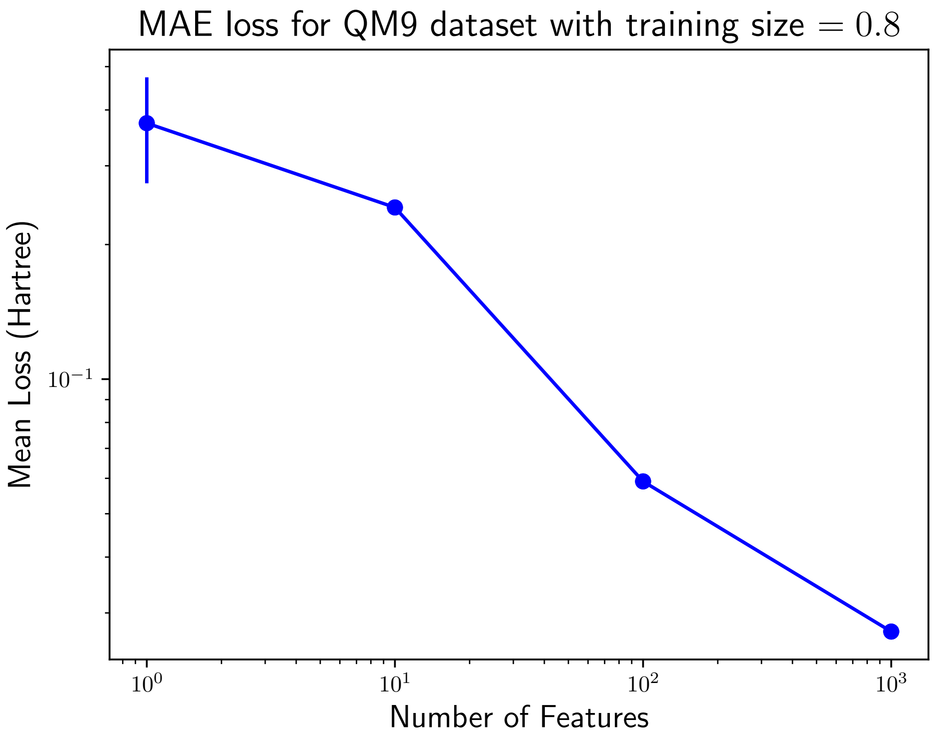

Figure 2: MAE as a function of feature size at training size 0.8.

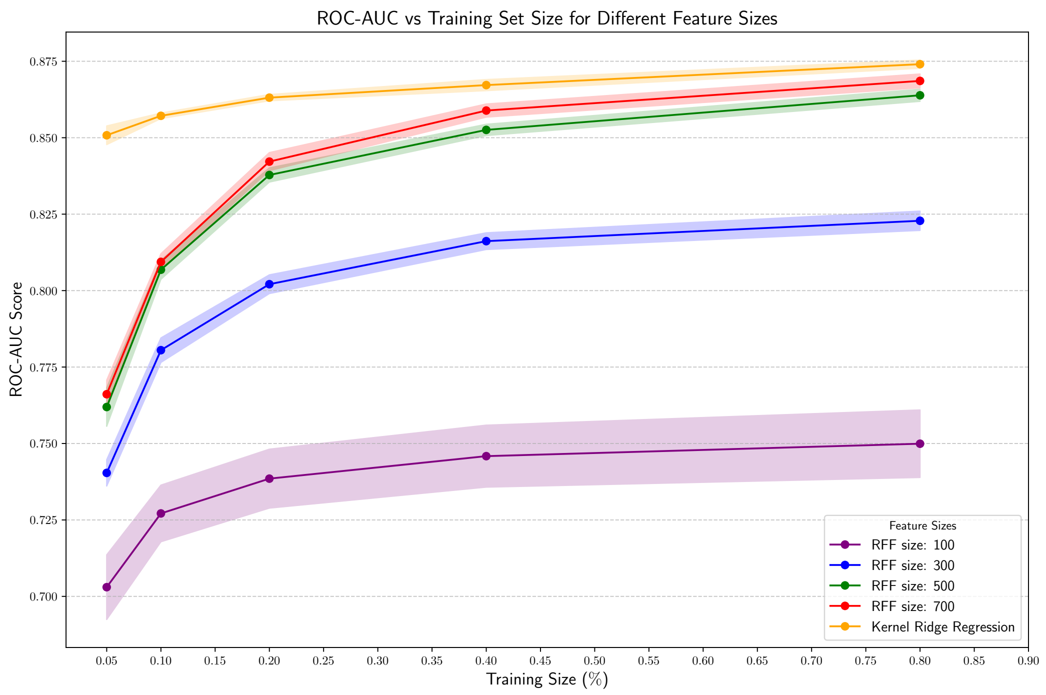

Figure 3: Accuracies of models trained with different features, as a function of training set size. The standard error is shown as the shaded region around the lines.

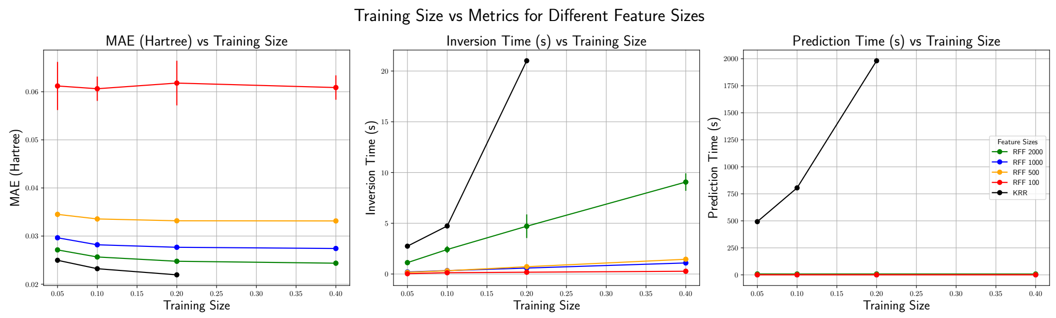

Figure 4: Plots of the MAE, Inversion Time, and Prediction time for the QM9 dataset with varying RFF feature sizes as well as a regular KRR approach.

4.2 RFF Performance Metrics Across Training Set Size

In Figure 3, the response of the classification error to the training set size is plotted at different numbers of RFFs. The regular KRR predictions are also included for comparison. The RFF models perform much worse than the regular KRR model at small training set sizes but have higher learning slopes. The regular KRR model performs quite well at even small training set sizes. At higher training set sizes, the higher feature sizes approach the performance of the KRR model. In all models the shape of the learning curve is the same as that of the learning curve with respect to the feature size.

Learning curves for QM9 are shown in Figure 4(a). Only three points were collected for the regular KRR model due to time constraints. The slopes of the learning curves are very similar except for feature size 100, but the offsets are well separated. This illustrates the effect of the feature size on the regression problem; the ability of the model to learn from more data is not hindered by the number of random features. While the error is higher with the RFF models, the computational savings are immense as demonstrated in Figures 4(b) and (c). The time for kernel inversion is seen to scale linearly for the RFF models, with higher numbers of features leading to higher scaling. The scaling of the computational cost of the matrix inversion for the regular KRR model is approximately cubic. Also notable is the differences in prediction time. The magnitude of the time required for generating the test kernel in the regular KRR model is so high that the RFF model predictions are effectively instantaneous in comparison. These findings highlight the power of the RFF approximation: competitive errors can be reached at a fraction of the computational cost.

5 Conclusion

This study examined the trade-offs between Random Fourier Features (RFFs) and traditional Kernel Ridge Regression (KRR) in terms of performance and computational efficiency. Experiments on the MAGIC dataset for classification and the QM9 dataset for regression revealed several key findings.

Increasing the number of RFFs consistently improved accuracy. For classification, error decreased logarithmically, while for regression, the relationship was nearly linear. Larger feature sizes also reduced variance in error, reflecting more robust feature representations. In terms of training set size, models trained on larger datasets generalized better. Regular KRR performed better with smaller datasets due to its exact kernel computations, but RFFs achieved comparable performance with larger datasets, demonstrating scalability.

The standout advantage of RFFs was their computational efficiency. While KRR scales roughly cubically with the number of data points, RFFs scale linearly with the number of features, significantly reducing training and prediction times. These savings make RFFs highly suitable for large-scale or real-time applications.

Despite slightly lower accuracy at small feature sizes, RFFs offer a practical and scalable alternative for kernel-based methods when computational constraints are critical. This makes them particularly effective for large datasets that would otherwise be computationally prohibitive with traditional approaches. Overall, this study highlights the potential of RFFs to balance accuracy and efficiency, enabling kernel-based models to scale effectively in real-world machine learning applications.

References

[1] Thorsten Joachims. Training linear svms in linear time. In *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '06, page 217–226, New York, NY, USA, 2006. Association for Computing Machinery.

[2] Michael C Ferris and Todd S Munson. Interior-point methods for massive support vector machines. *SIAM Journal on Optimization*, 13(3):783–804, 2002.

[3] Shai Shalev-Shwartz, Yoram Singer, and Nathan Srebro. Pegasos: Primal estimated sub-gradient solver for svm. In *Proceedings of the 24th international conference on Machine learning*, pages 807–814, 2007.

[4] Dennis DeCoste and Dominic Mazzoni. Fast query-optimized kernel machine classification via incremental approximate nearest support vectors. In *ICML*, pages 115–122, 2003.

[5] Haim Avron, Michael Kapralov, Cameron Musco, Christopher Musco, Ameya Velingker, and Amir Zandieh. Random fourier features for kernel ridge regression: Approximation bounds and statistical guarantees. In *International conference on machine learning*, pages 253–262. PMLR, 2017.

[6] Yuchen Zhang, John Duchi, and Martin Wainwright. Divide and conquer kernel ridge regression: A distributed algorithm with minimax optimal rates. *The Journal of Machine Learning Research*, 16(1):3299–3340, 2015.

[7] Ahmed Alaoui and Michael W Mahoney. Fast randomized kernel ridge regression with statistical guarantees. *Advances in neural information processing systems*, 28, 2015.

[8] Cameron Musco and Christopher Musco. Recursive sampling for the nystrom method. *Advances in neural information processing systems*, 30, 2017.

[9] Ali Rahimi and Benjamin Recht. Random features for large-scale kernel machines. *Advances in neural information processing systems*, 20, 2007.

[10] Salomon Bochner. Vorlesungen über fouriersche integrale. *(No Title)*, 1948.

[11] W. Rudin. *Fourier Analysis on Groups*. Dover Books on Mathematics. Dover Publications, 2017.

[12] L. C. Blum and J.-L. Reymond. 970 million druglike small molecules for virtual screening in the chemical universe database GDB-13. *J. Am. Chem. Soc.*, 131:8732, 2009.

[13] M. Rupp, A. Tkatchenko, K.-R. Müller, and O. A. von Lilienfeld. Fast and accurate modeling of molecular atomization energies with machine learning. *Physical Review Letters*, 108:058301, 2012.

[14] James Mercer. Xvi. functions of positive and negative type, and their connection the theory of integral equations. *Philosophical transactions of the royal society of London. Series A, containing papers of a mathematical or physical character*, 209(441-458):415–446, 1909.

[15] R. Bock. MAGIC Gamma Telescope. UCI Machine Learning Repository, 2004. DOI: https://doi.org/10.24432/C52C8B.

[16] Raghunathan Ramakrishnan, Pavlo O Dral, Matthias Rupp, and O Anatole Von Lilienfeld. Quantum chemistry structures and properties of 134 kilo molecules. *Scientific data*, 1(1):1–7, 2014.

[17] Vladimir Vapnik. Statistical learning theory. *John Wiley & Sons google schola*, 2:831–842, 1998.

6 Contributions

- **Scott:**
  - Wrote Kernel Ridge Regression script with grid search hyperparameter optimization.
  - Wrote the Background section of the report.
  - Contributed to the Results section.
- **Richard:**
  - Wrote the script to train models across different feature sizes.
  - Trained models and created learning curves.
  - Wrote the Methods and Results section of the report.
- **Karthik:**
  - Wrote the Random Fourier Feature algorithm.
  - Wrote the Introduction section of the report.
  - Generated the figures.
  - Wrote the Conclusion section.

All other works not listed above have been equally distributed across the members of the team.