Pack only the essentials: Adaptive dictionary learning for kernel ridge regression

Daniele Calandriello, Alessandro Lazaric, Michal Valko

Motivation
- Kernel regression is versatile and accurate
- Strong accuracy guarantees but poor scalability
  \( O(n^2) \) time \( \mathbb{C}(n^3) \) space (a number of samples)
- Current limitation: Many approximate schemes are either not scalable or not accurate
  – We propose an incremental approximation scheme for kernel regression with complexity and error guarantees depending on kernel structure

Kernel Ridge Regression (KRR)

The setting (fixed-design)
- Dataset \( \mathcal{D} = \{x_i, y_i\}_{i=1}^n \)
  - arbitrary \( x_i, y_i \)
  - \( y_i \sim f(x_i) + \varepsilon \)
- Kernel function \( K: \mathcal{X} \times \mathcal{X} \to \mathbb{R} \)
- Kernel matrix \( K_n \in \mathbb{R}^{n \times n} \), with \( [K]_{i,j} = K(x_i, x_j), i,j \leq n \)

Kernel regression
- Objective (after \( t \) samples)
  \( \hat{y}_t = \arg \min \{ \| y - K \hat{\omega} \|^2 + \| \hat{\omega} \|^2 \} \)
- Closed-form solution
  \( \hat{y}_t = (K_t + \beta I)^{-1} y_t \)
- On-sample risk
  \( \mathbb{R}(y_i) = \mathbb{R}_n[y_i] = \mathbb{R}_n(y_t - K \hat{\omega}_t) \)

Nyström Approximation

Subsampling
1. Select a subset (dictionary) \( \mathcal{Z}_n \) of \( m \) representative samples
2. Constructs a sparse matrix \( S_m \) to select and reweight the columns associated with the points in \( \mathcal{Z}_n \)

Low-Rank Approximation
3. Compute approximate, low-rank matrix \( \hat{K}_n = \text{CW}^{-1} \) as
   \[ \hat{K}_n = \text{CW}^{-1} = \text{KS}_m \text{S}_m^T (\text{S}_m \text{KS}_m + \beta I)^{-1} \text{S}_m \text{KS}_m \]

Efficient Solution
4. Compute approximate solution
   \( \hat{y}_n = \hat{K}_n \beta I \hat{y}_n = \frac{1}{\gamma} (y - C(C^T C + \beta W)^{-1} C^T y) \)

Scalability now depends on \( m \)

Space \( O(m^2) \rightarrow O(nm) \), Time \( O(m^3) \rightarrow O(m^2 + m^3) \)

Problems:
- How to choose the sampling distribution?
- How to choose \( m \)?

References
[Rudi et al.(2015)] A. Rudi, R. Camoriano, and L. Rosasco. Less is more: The setting (fixed-design)

Kernel Ridge Leverage Scores (RLS) Sampling for KRR

Definition 1. Given a kernel matrix \( K_n \in \mathbb{R}^{n \times n} \), define
   \[ \gamma_{i,j} = \left\| (y - K \hat{\omega}_t)_{i,j} \right\| \]

Proposition 1 (Alaoui, Mahoney, 2015). Let \( \gamma_i \) be the leverage, \( \delta \) the confidence. If the regularized Nyström approximation \( K_n \) is computed using the sampling distribution \( (p_{i,j})_n \), and at least
   \[ m \geq \frac{1}{\gamma (1 + \delta) \log(n)} \]
   columns, then with probability \( 1 - \delta \)
   \[ \mathbb{R}(y_i) \leq (1 + \delta) \mathbb{R}_n(y_i) \]

SQUEAK

Lemma 1. Assume that the dictionary \( \mathcal{Z}_n \) induces a \( \gamma \)-approx. \( K_n \), and let \( \mathcal{S}_n \) be constructed by adding \( \gamma \) copies of \( \mathcal{Z}_n \) to the selection matrix. Then, denoting \( n = (1 + \delta)(1/1 - \delta), \) for all \( i \) such that \( i \in \mathcal{Z}_n \cup \{1, \ldots, n\} \),
   \[ \gamma_{i,j} = \frac{1}{\gamma (1 + \delta) \log(n)} \]

is an \( \alpha \)-approximation of the RLS \( \gamma_{i,j} \), that is \( \gamma_{i,j} / \alpha \leq \gamma_{i,j} \).

SQUEAK

Input: Dataset \( \mathcal{D} \), regularization \( \gamma, \beta, \tau \)
Output: \( \hat{K}_n, \hat{\omega}_n 

1. Initialize \( \hat{y}_n \) as empty, \( p_{i,j} = 0 \)
2. for \( t = 1, \ldots, n \) do
3. Receive new column \( \mathcal{Z}_t \)
4. Compute \( \alpha \)-approx. RLS \( (\tau_t) \in Z_t \cup \{1, \ldots, n\} \), using \( Z_{t-1} \), \( \mathcal{Z}_t \), and Eq. 4
5. Set \( p_{\tau_t} = \max \{ p_{\tau_t}, p_{\tau_t + 1, \tau_t + 1/2} \} \)
6. Initialize \( \hat{K}_{n-1} \)
7. for all \( i \in \{ 1, \ldots, n \} \) do
8. Set \( Q_{i,j} \leftarrow [1 + \delta \{ j \in Z_i \}] \)
9. if \( Q_{i,j} \neq 0 \) then
10. \( \gamma_{i,j} = \mathbb{R}(y_i) / \alpha \)
11. Add \( Q_{i,j} \) copies of \( \left( K_{i,j}, p_{i,j} \right) \) to \( \mathcal{Z}_n \)
12. end if
13. end for
14. \( \mathcal{S}_n \leftarrow \mathcal{S}_n + Q_{i,j} \)
15. Add \( Q_{i,j} \) copies of \( \left( K_{i,j}, p_{i,j} \right) \) to \( \mathcal{Z}_n \)
16. Compute \( \hat{K}_n \) using \( \hat{K}_n \) and \( \hat{y}_n \)
17. end for

Theorem 1. Let \( \alpha = \frac{1}{n} \) and \( \gamma > 1 \). For any \( 0 \leq \delta \leq 1 \), and \( 0 \leq \delta \leq 1 \), if we run SQUEAK with \( \tau \) and \( \mathcal{Z}_n \), then \( 1 - \delta, \) for all \( t \in [n] \)

1. \( K_n \) computed with \( \mathcal{Z}_n \) is a \( \gamma \)-approximation of \( K_n \)
2. \( \mathbb{R}(y_i) \leq (1 + \delta) \mathbb{R}_n(y_i) \)
3. \( \mathbb{R}(y_i) \leq (1 + \delta) \mathbb{R}_n(y_i) \)

Beyond sequential KRR

What if we run SQUEAK simply to approximate \( K_n \)?
- Only need to compute RLS for points in \( \mathcal{I}_n \) never recompute after dropping
- Never construct the whole \( K_n \), subquadratic runtime \( O(m^2 + m^3) \rightarrow O(m^2 + m^3) \)
- Store points directly in the dictionary
- \( O((m^2 + m^3) \rightarrow O(n^2) \), constant in \( m \), simple pass over the dataset (streaming)
- Extend DICT-UPDATE (add point to dictionary) to DICT-MERGE (add dictionary to dictionary)
- Distributed SQUEAK, multiple nodes operate in parallel, without sharing memory
- RLS sampling preserves the projection on \( K_n \)’s range
- RLS sampling preserves well the projection on \( K_n \)’s range

SQUEAK provides strong guarantees for many Kernel problems (randomized design KRR, Kernel PCA, Kernel k-means)

Pros:
- \( \beta \) and \( \alpha \) trade off accuracy and speed/time cost
- The formulation of \( K_n \) is not incremental

Cons:
- The time bottleneck is computing interme-
diate KRR solutions \( O((m^2 + m^3) \rightarrow O(n^2) \)
- Still potentially constructs the whole matrix to compute KRR, single pass over matrix but not dataset.