# Space and Time Efficient Kernel Density Estimation in High Dimensions

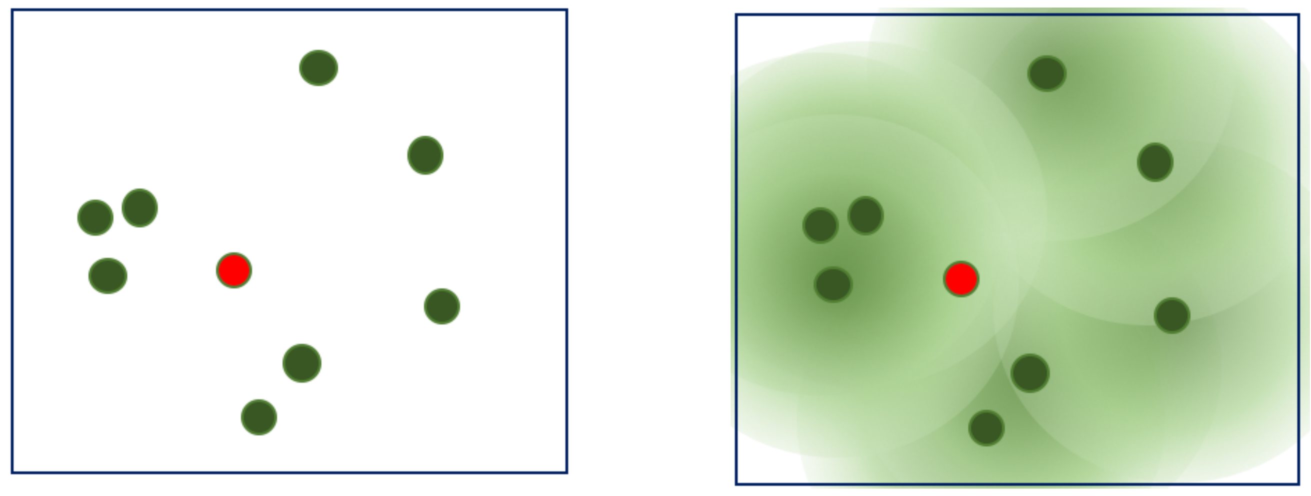

Arturs Backurs — TTIC

Piotr Indyk — MIT

Tal Wagner — MIT


## Kernel Density Estimation

**Problem**: Given a dataset $x_1, \dots, x_n \in \mathbb{R}^d$, estimate density at a query point $y \in \mathbb{R}^d$.



**Method**: Define a similarity measure ("*kernel*"): $k: \mathbb{R}^d \times \mathbb{R}^d \to [0,1]$

Define the **Kernel Density Estimation** as:

$$KDE_X(y) = \frac{1}{n}\sum_{i=1}^{n} k(x_i, y)$$

**Examples of popular kernels:**

"Exponential": $k(x,y) = \exp\left(-\frac{\|x-y\|_2}{\sigma}\right)$

"Laplacian": $k(x,y) = \exp\left(-\frac{\|x-y\|_1}{\sigma}\right)$

"Gaussian": $k(x,y) = \exp\left(-\frac{\|x-y\|_2^2}{\sigma}\right)$

"Cauchy": $k(x,y) = \frac{1}{1+\|x-y\|_2^2}$

## Methods for fast KDE

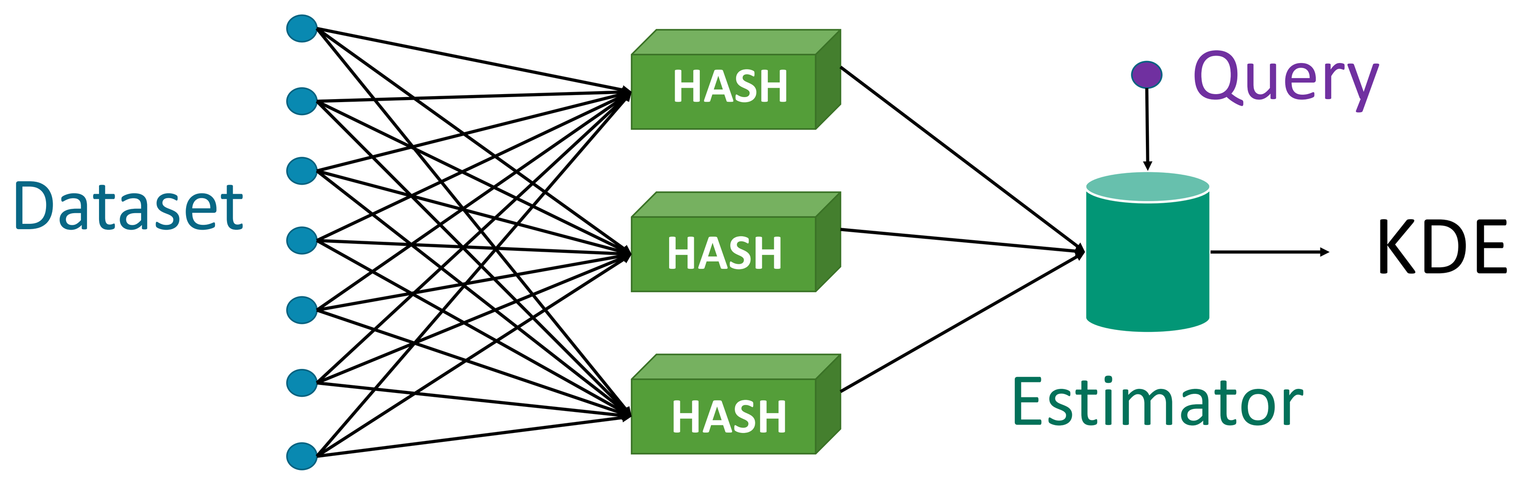
**Goal**: Compute a $(1 \pm \varepsilon)$ approximation of $KDE_X(y)$.

**Assumption**: $KDE_X(y) \geq \tau$ for some $\tau > 0$. *(i.e.: Query not too unrelated to dataset)*

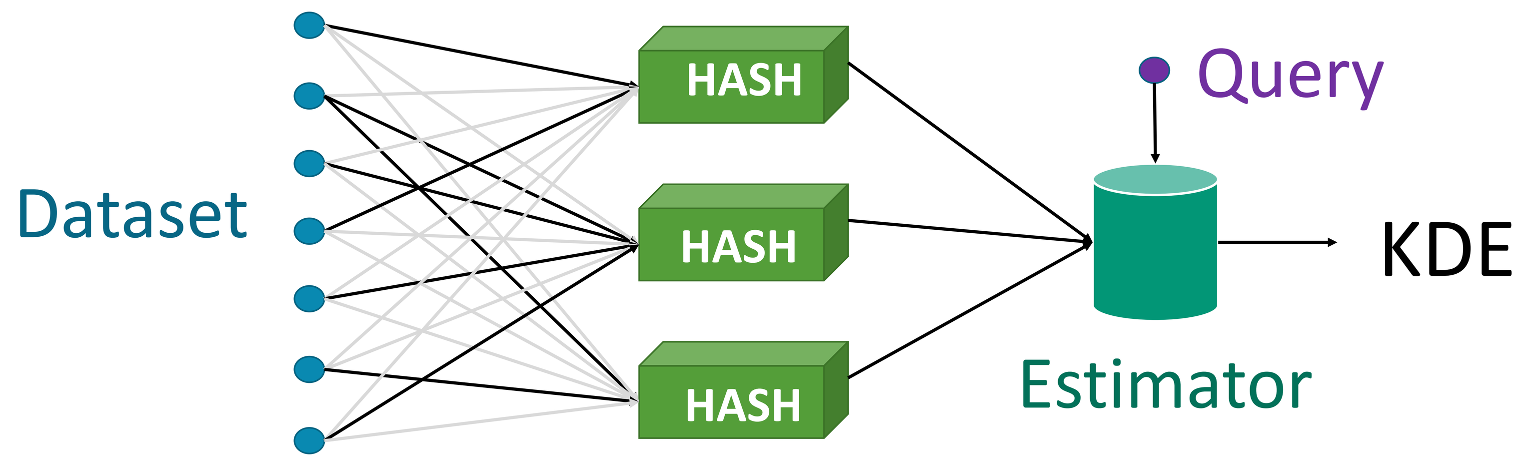
| Method | Query time | Space usage & preprocessing time |
|---|---|---|
| Exact computation | $O(n)$ | None |
| Vanilla uniform sampling | $O(1/(\tau \cdot \varepsilon^2))$ | $O(1/(\tau \cdot \varepsilon^2))$ |
| Hashing-based estimators (HBE) [Charikar & Siminelakis 2017] | $O(1/(\sqrt{\tau} \cdot \varepsilon^2))$ | $O(1/(\tau\sqrt{\tau} \cdot \varepsilon^4))$ |
| **This work** (modified HBE) | $O(1/(\sqrt{\tau} \cdot \varepsilon^2))$ | $O(1/(\tau \cdot \varepsilon^2))$ |

Best of both worlds

**HBE** [CS'17]: Use **Locality-Sensitive Hashing (LSH)** in preprocessing for **importance sampling** in querying.

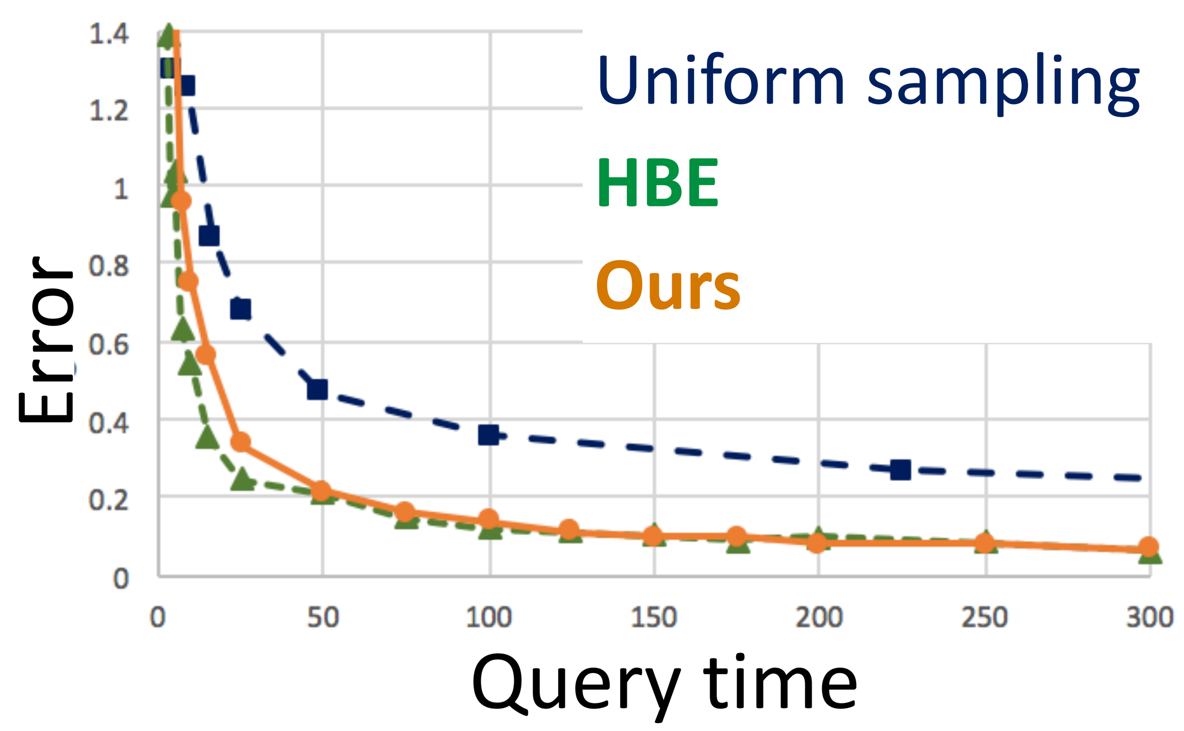


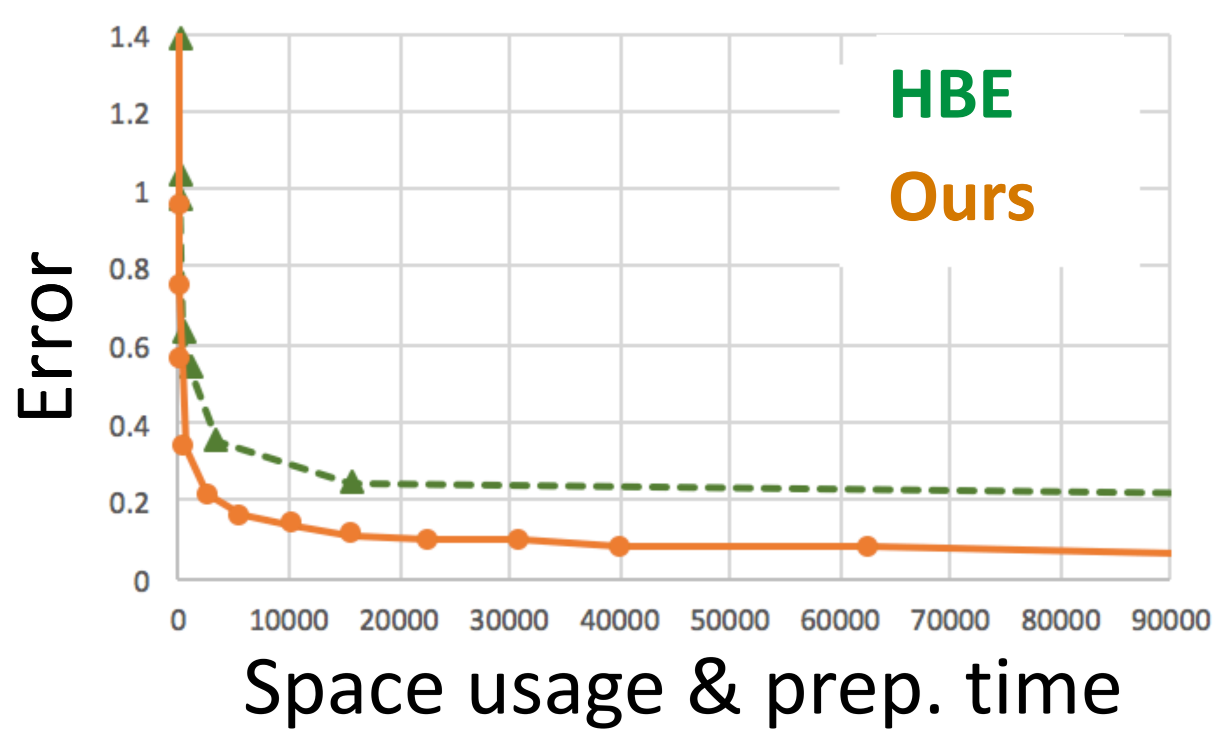

**Our space-efficient HBE**: Introduce random drop-out in the hashing step.




**Experiments:** Our query time is similar to HBE, with better space usage and preprocessing time.

Accuracy is better than uniform sampling on some datasets [Siminelakis et al. 2019]





**References:** ▹ M. Charikar, P. Siminelakis. **Hashing-based-estimators for kernel density in high dimensions.** FOCS 2017 ▹ P. Siminelakis, K. Rong, P. Bailis, M. Charikar, P. Levis. **Rehashing kernel evaluation in high dimensions.** ICML 2019