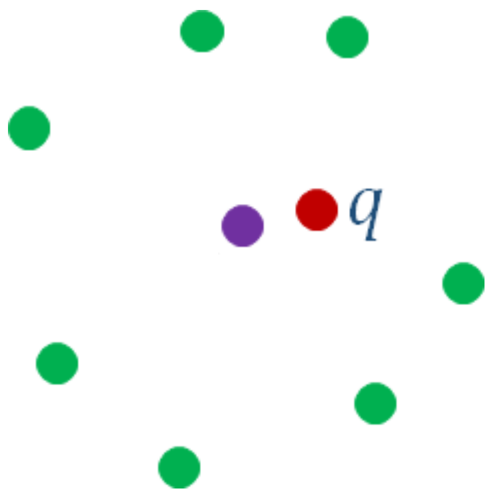
Learning Space Partitions for Nearest Neighbor Search

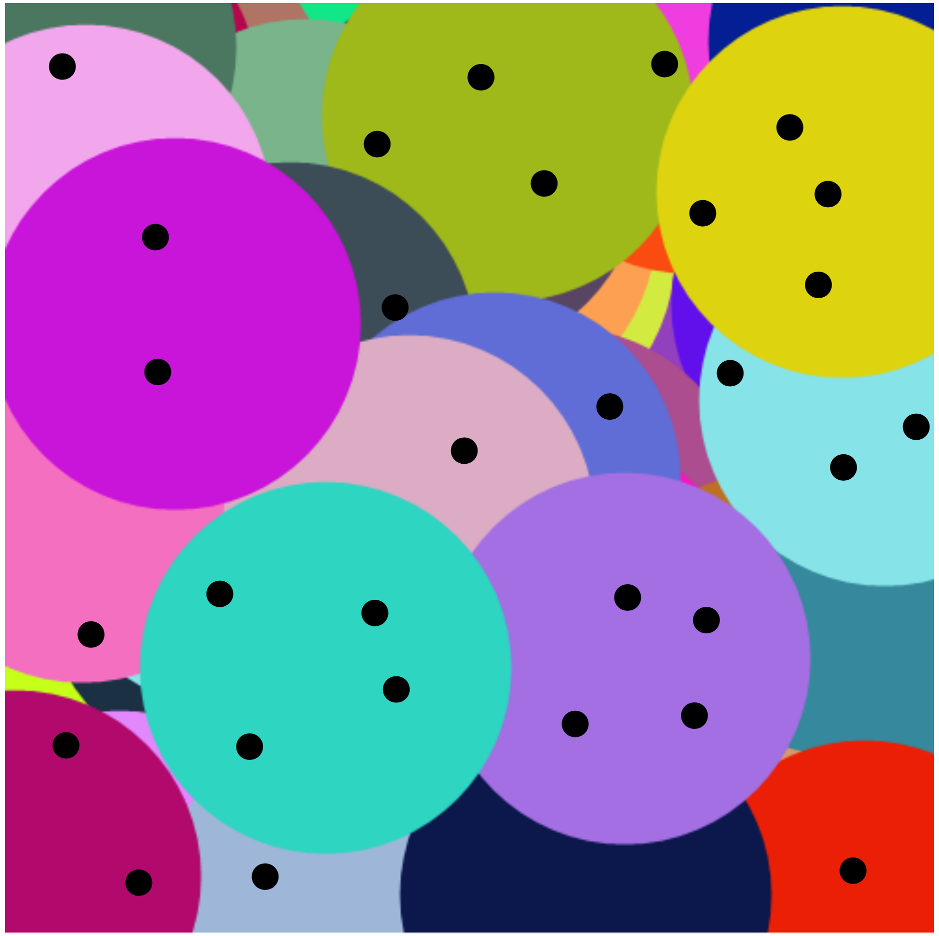
Yihe Dong	Piotr Indyk	Ilya Razenshteyn	Tal Wagner
MSR	MIT	MSR	MIT

Nearest Neighbor Search

- Given:
  - Dataset of points in $\mathbb{R}^d$.
- Query:
  - q in $\mathbb{R}^d$.
- Goal:
  - k-nearest neighbors from dataset.



Method: Space Partitions of $\mathbb{R}^d$

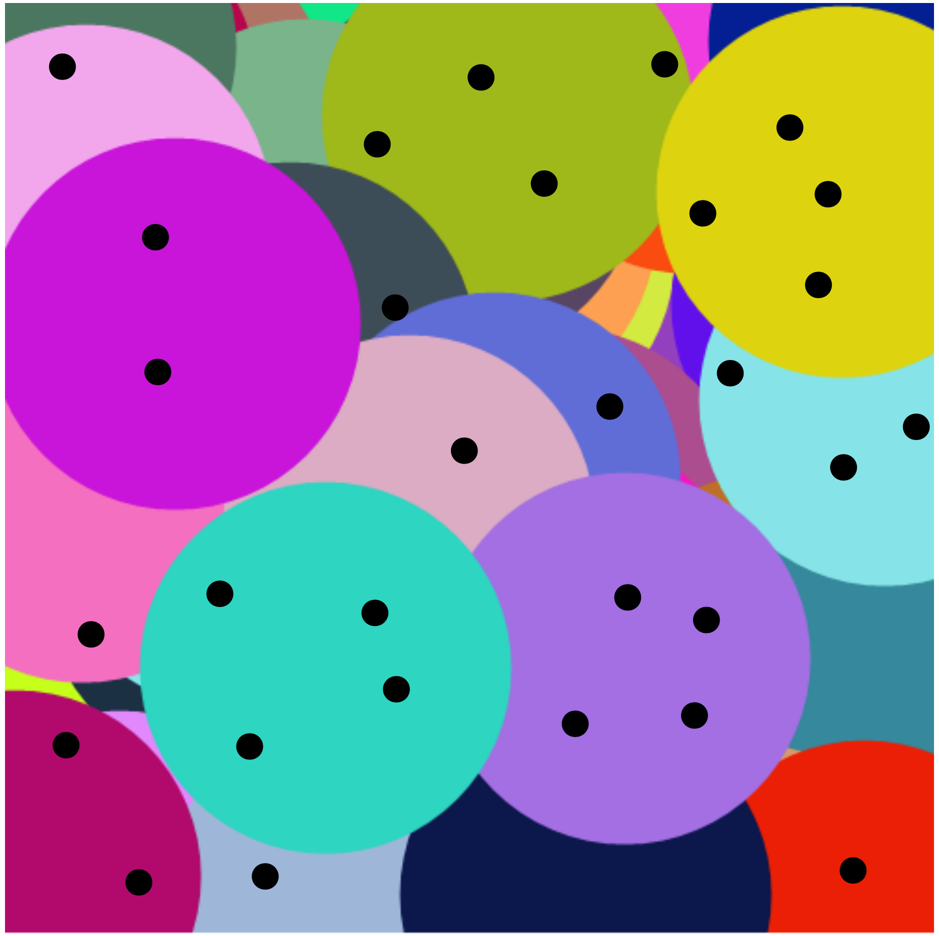


Advantages:

- Sublinear query time
  - Compute distance from query to a subset of candidate data points
- Distributed computation
  - Put each bin on different machine

Space Partition Desiderata

- Want a partition of $\mathbb{R}^d$ that:
  - Returns accurate nearest neighbors
  - Approximately balanced
    - w.r.t. data points
  - Algorithmically simple



Methods for Space Partitions

- Data independent:
  - Classical Locality-sensitive hashing (LSH)

- Data dependent:
  - Data dependent LSH
  - Quantization (k-means)
  - **Supervised** hyperplane partitions

- **Our goal: Use modern supervised learning (like neural networks) to learn better space partitions**

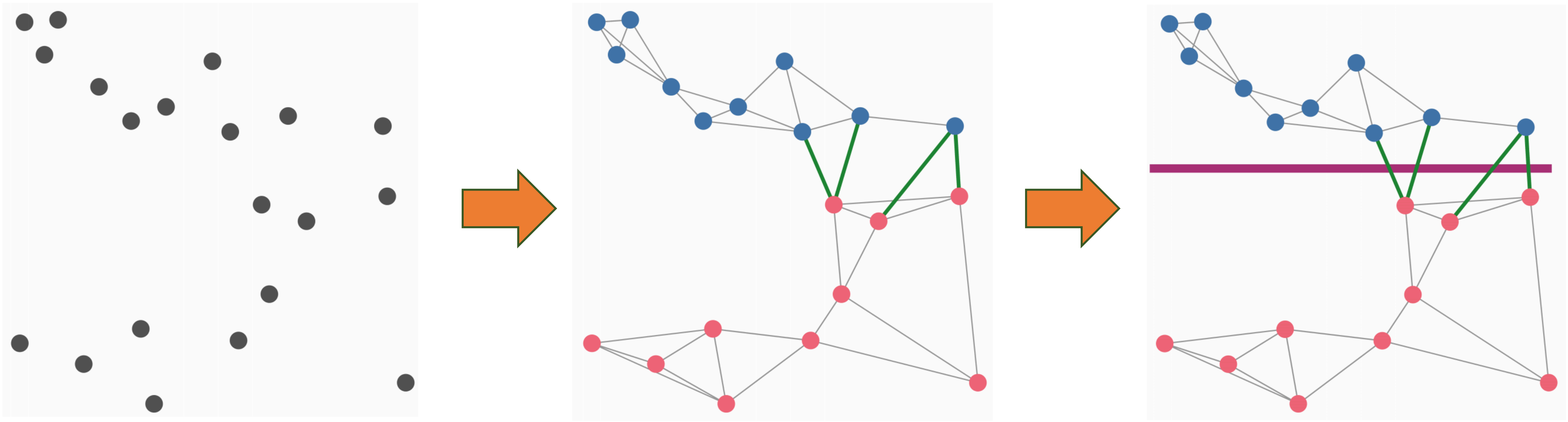
Our Contribution

- New method to partition $\mathbb{R}^d$
- Two stage process:
  1. Combinatorial graph partitioning
  2. Supervised learning
- Empirically better than prior methods for nearest neighbor search

We use **KaHIP** (Sanders and Schultz 2013)

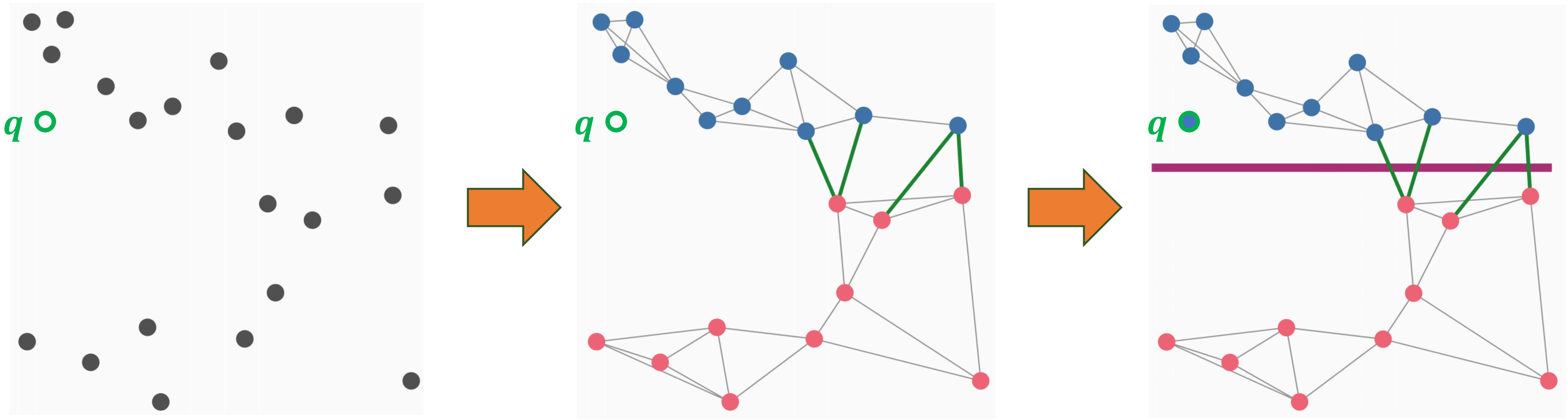
We use **small neural networks** ("Neural LSH")

Our Method: Preprocessing

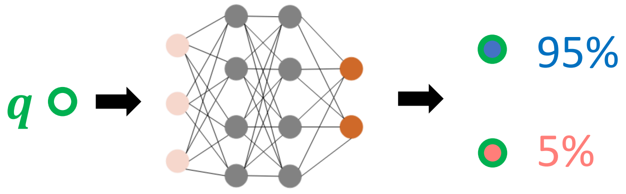


- Create k-NN graph of dataset
- Find balanced partition of graph
- **Train learning model** to generalize partition from graph nodes to all of $\mathbb{R}^d$

Our Method: Query



- Run **inference** on query to classify into bin, or to get ranking of likely bins
- Search for nearest neighbors in highest ranking bins

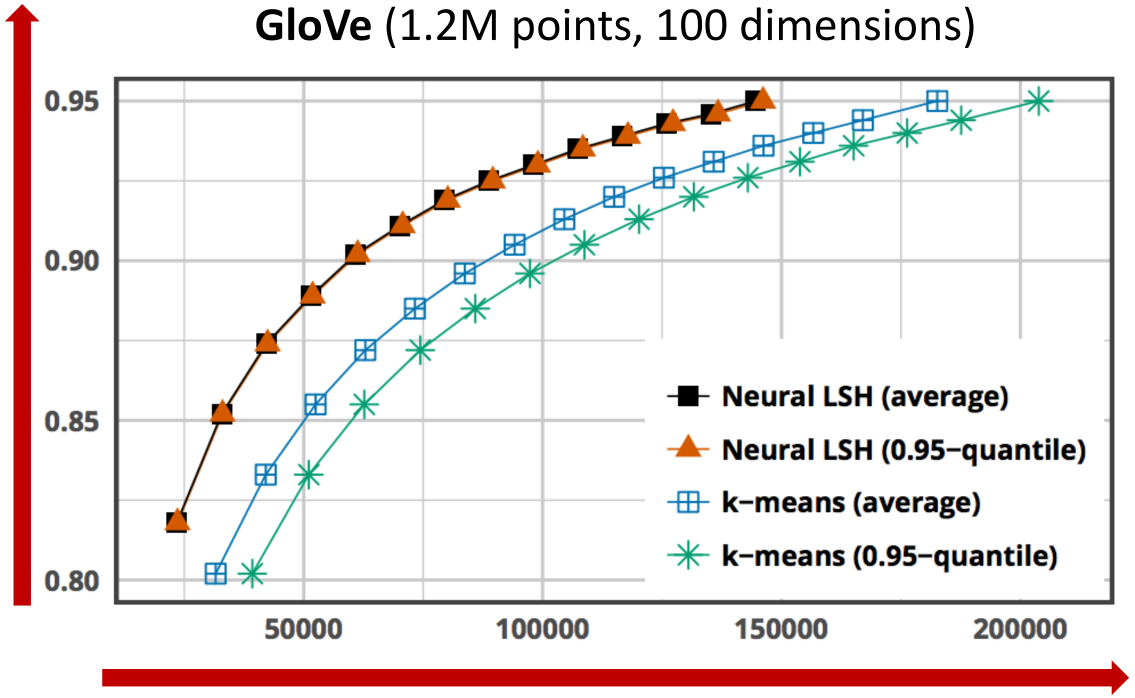


Select Experimental Results

- Partition into **256** bins

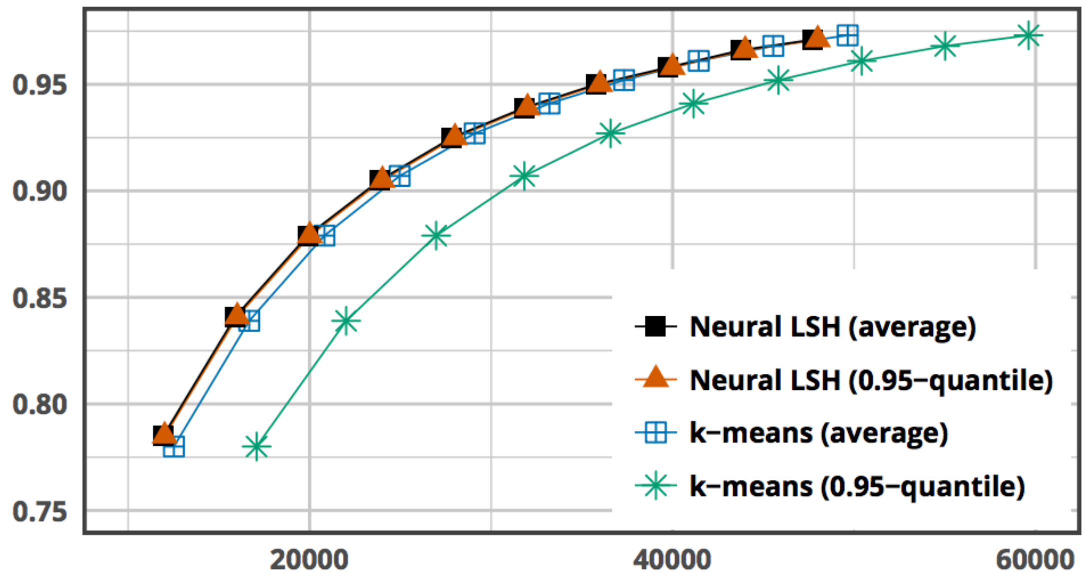
GloVe (1.2M points, 100 dimensions)

k-NN accuracy (k=10)



Number of distance computations *("candidates")*

SIFT (1M points, 128 dimensions)



Code on GitHub

Thank you