Scalable k-Means Clustering for Large k via Seeded Approximate Nearest-Neighbor Search

Jack Spalding-Jamieson (Independent)
Eliot Robson (UIUC)
Da Wei Zheng (UIUC)

Problem

k-means on large high-dimensional datasets is slow for big k!

Existing Approach: LLoyd's Algorithm

1. **Initialize** A set of k centers C by uniform sampling from P.
2. **Assign** each point P to its closest center. Bottleneck
   Terminate if no changes.
3. **Recompute** cluster centers C_i as mean of assigned points. Go to step 2.

Easy Idea: ANNS for Assignments

1. **Initialize** A set of k centers C by uniform sampling from P.
2. **Build** an in-memory ANNS data structure over C.
3. **Assign** each point P to its approximate closest center. Terminate if no changes.
4. **Recompute** cluster centers C_i as mean of assigned points. Go to step 2.

Our Solution: Seeded Approximate Nearest-Neighbor Search (SANNS)

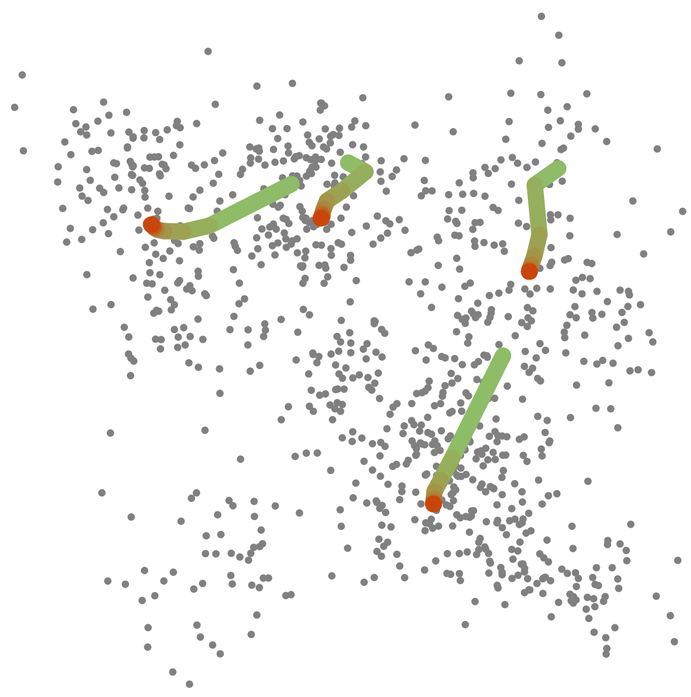
- For each query, given initial guesses (called seeds) for candidate nearest neighbors.
- Learning-augmented form of ANNS.
- More appropriate family of problems to study for k-means acceleration: **Seeds come from previous iteration.**
- Present a framework of solutions to SANNS that we call seeded search-graphs.
- Result: Seeded search-graphs for k-means clustering (SHEESH).

The Algorithm

Input: $P \subset \mathbb{R}^d$, centers $C \subset \mathbb{R}^d$, $|C| = k$, previous multi-assignments $S : P \to 2^C$, previous search-graph data structure D
Build: Build a new search-graph data structure D' by using D.
Reassign:
for each chunk U of $O(k)$ points in P (in parallel) **do**
  Group the points of U into roughly-correlated groups.
  Randomly project each group into $\mathbb{R}^1$, and sort the projected group.
  for each group G of U **do**
    for each point p of G, in the sorted order **do**
      Let q be the previous point.
      Use $S'(q) \cup S(p)$ as seeds.
      With all these seeds, compute the seeded approximate ~ 10 nearest centers of p using D', and save the results as $S'(p)$.
    end for
  end for
end for
Recompute: Compute the new centers C' as centroids.
Output: New centers C', new multi-assignments S', new search-graph data structure D'

Why does this work?

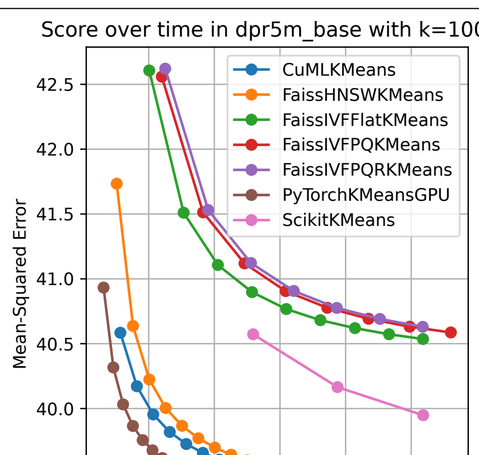
- Centroids slow down over time:



Datasets

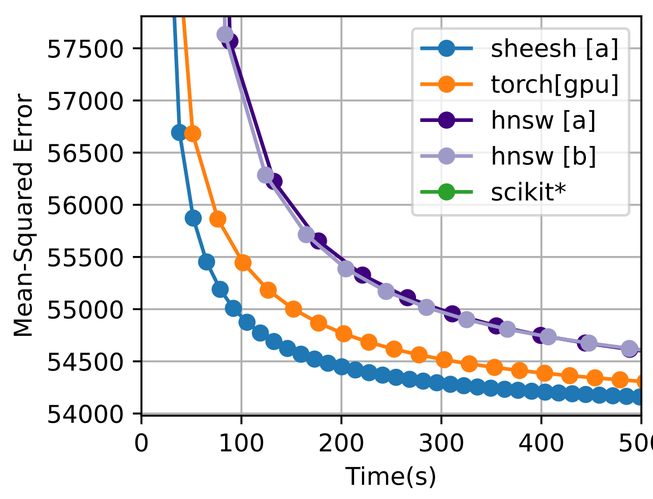
Dataset	Type	Dim	Points	Size
SIFT20M	Image	128	20 mil.	10.24 GB
~~Text2~~Image10M	Image	200	10 mil.	8.00 GB
DPR5M	Text	768	5 mil.	15.36 GB

Results with ANNS Methods

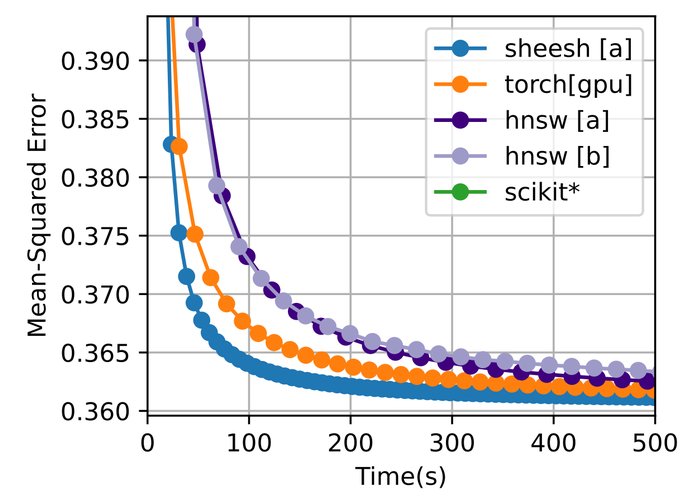


Comparison of "Easy Idea" with ANNS methods in FAISS versus baselines PyTorch, CuML, and Scikit.

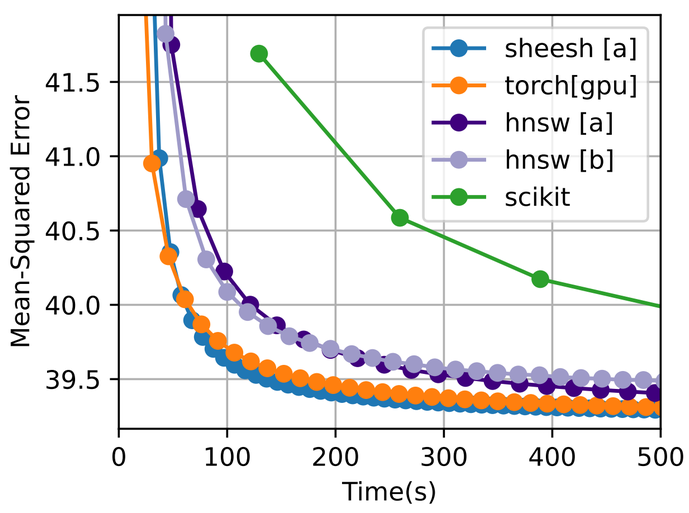
Results with Our Seeded ANNS Method (SHEESH)



SIFT20M, $k = 10000$



~~Text2~~Image10M, $k = 50000$



DPR5m, $k = 100000$