Large-scale NN Search: Graph-based Approaches

Dr. Wan-Lei *Zhao*

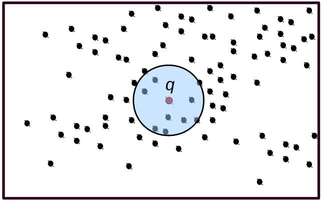
Aug. 29 2022

Contact: wlzhao@xmu.edu.cn

Outline

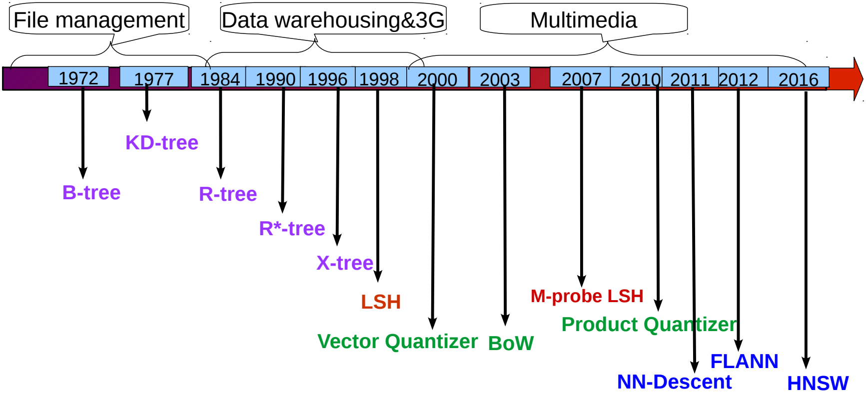
Nearest Neighboor Search: the problem

- Given a set of samples S in a metric space m and a query sample $q \in R^d$
- Task: find nearest neighbors from set S for sample q



- In most of the practices, the algorithm should be able to return k nearest neighbors (at least the top one)

A Glimpse over NNS History (1)



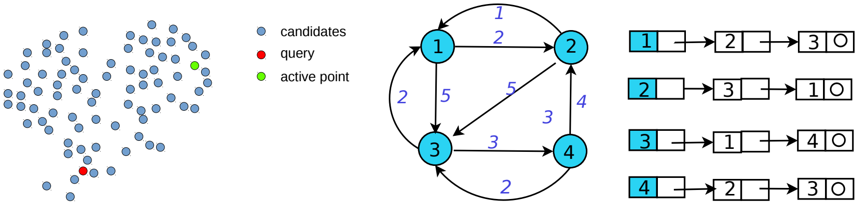
- In the whole 90s and early 00s, researchers were working on "trees"
- The introduction of Web 2.0 changes the culture

A Glimpse over NNS History (2): Divide & Conquer

Low & Dense **<u>KD-tree, R-tree</u>**	**Low & Sparse** **<u>KD-tree, R-tree</u>**
High & Dense **<u>???</u>**	**High & Sparse** **<u>Inverted file</u>**

- NNS on low dimensional data is solved
- NNS on sparse data is solved
- No effective solution for high and dense space

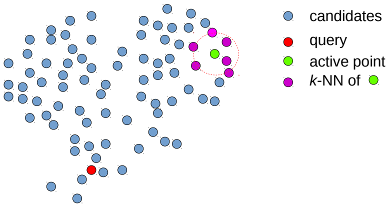
Graph-based NN Search: the idea



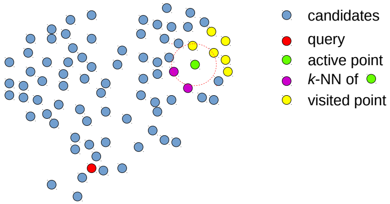
(a) Initial state (b) Directed graph (c) 2-NN graph of (b)

- Given query and the candidate set
- Sample a candidate as seed randomly
- A k-NN graph is used as a routing table
  - Expand neighbors of active points iteratively
- Climb to the query as much as we can

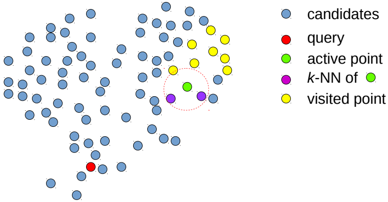
Graph-based NN Search: the procedure (1)



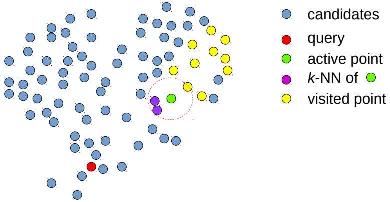
(a) Step-1



(b) Step-2

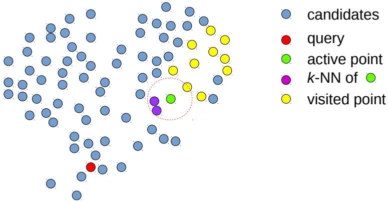


(c) Step-3

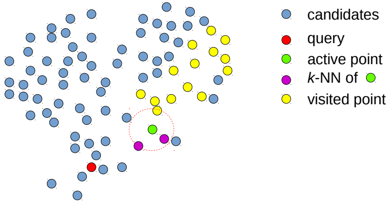


(d) Step-4

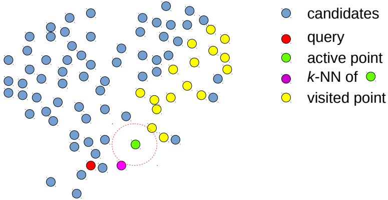
Graph-based NN Search: the procedure (2)



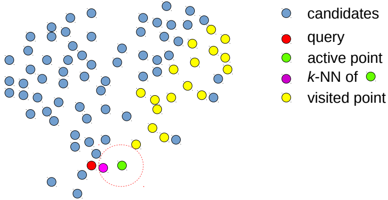
(d) Step-4



(e) Step-5



(f) Step-6



(g) Step-7

Outline

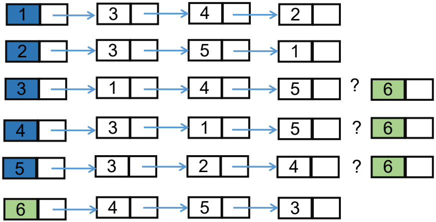
1. Overview about Graph-based NN Search
2. Online Approximate k-NN Graph construction
3. k-NN Graph Merge
   - Symmetric Merge
   - Joint Merge

Online k-NN Graph: the motivation

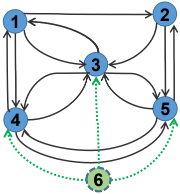
1. To support the NN Search, k-NN graph is required
2. Existing k-NN Graph construction works on static dataset
3. In practice, dynamic update (insert/delete) on the dataset should be allowed
4. A dynamic k-NN graph for NN search is required

Facing the similar issue as traditional relational database!

Online k-NN Graph: the idea



(a) 3-NN lists



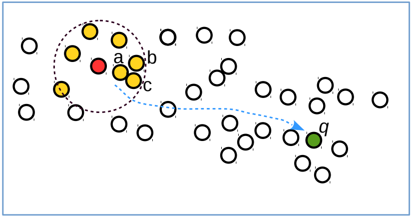
(b) 3-NN graph

1. There are 5 vertices in the graph
2. Vertex 6 is to be inserted
3. Idea: search over the graph and insert Vertex 6

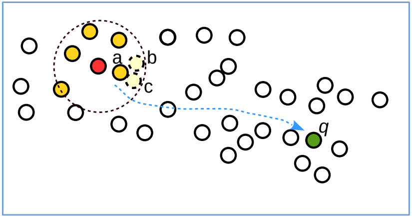
Online k-NN Graph: the major issues

1. Speed-up the search
2. Find out the k-NN list for the new vertex as complete as possible

Speed-up the search: Lazy Graph Diversification (1)

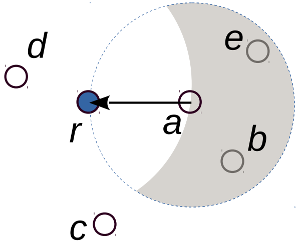


$\Downarrow$



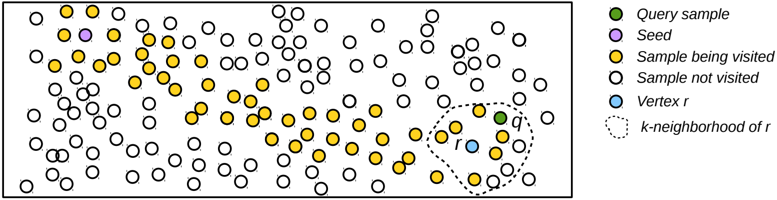
Avoid the comparison with ‘b’ and ‘c’, which are close to ‘a’

Speed-up the search: Lazy Graph Diversification (2)



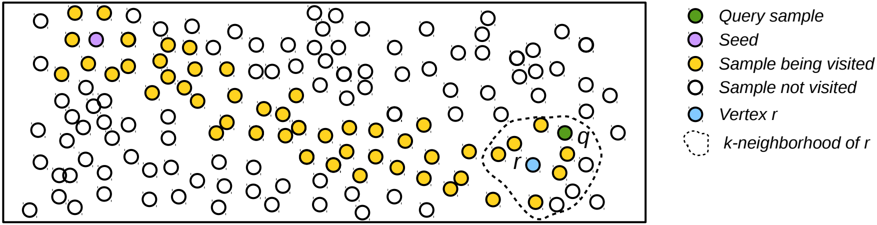
- Idea: ignore samples that have been occluded by others
  - Sample b and e are occluded by a
  - This idea is not new, but it is new that we do it online
  - Existing solutions require pair-wise comparison in r's neighborhood

Speed-up the search: Lazy Graph Diversification (3)



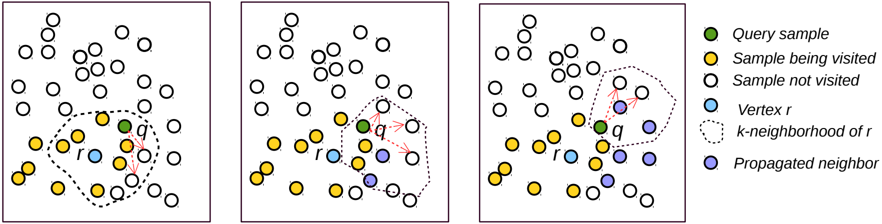
- Make use of the distances compuated during the hill-climbing
- No additional comparison is carried out in r's neighborhood
- The num. of comparisons is reduced by 50%

Recursive Neighborhood Propagation: motivation



- We want all the NNs when query reaches the target neighborhood
- There are some samples are not compared in the dashed circle
- They are most likely to be the neighbors of sample q
- Sample q should be introduced to these samples

Restricted Recursive Neighborhood Propagation



(a) Depth-1 (b) Depth-2 (c) Depth-3

- Sample q is introduced to the neighbors of visited samples'
- This could be done recursively for several rounds
- It turns out to be very cost-effective!!

Interpretation about these two Schemes

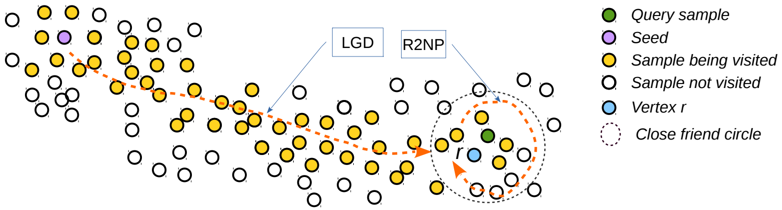


Figure: An illustration of "ballon" shape routing.

1. LGD (Lazy Graph Diversification): ignores close friends
2. R2NP (Restricted Recursive Neighborhood Propagation): introduced to friends' friends

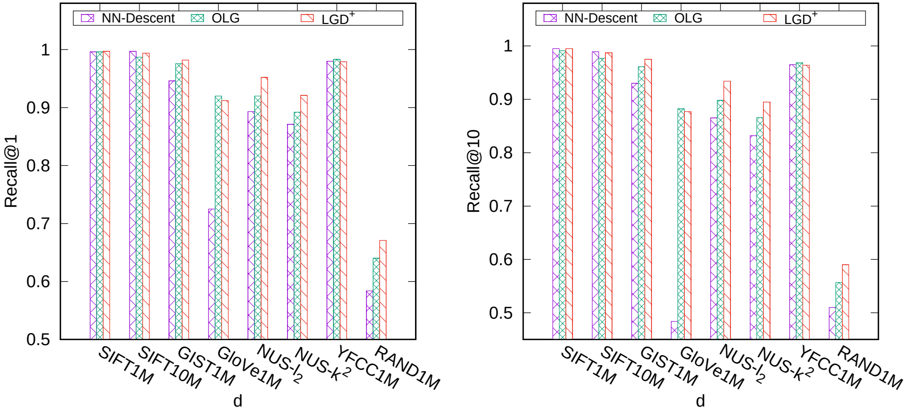
Datasets used for Evaluation

Table: Summary on Datasets

Name	n	d	# Qry	m($\cdot$)	Type
Rand100K	1×10^5	$3\sim100$	-	l_2	synthetic
Rand100K	1×10^5	$3\sim100$	-	l_1	synthetic
SIFT1M	1×10^6	128	1×10^4	l_2	SIFT
SIFT10M	1×10^7	128	1×10^4	l_2	SIFT
GIST1M	1×10^6	960	1×10^3	l_2	GIST
GloVe1M	1×10^6	100	1×10^3	*Cosine*	Text
NUSW	$22,660$	500	1×10^3	l_2	BoVW
NUSW	$22,660$	500	1×10^3	κ^2	BoVW
YFCC1M	1×10^6	128	1×10^4	l_2	Deep Feat.
Rand1M	1×10^6	100	1×10^3	l_2	synthetic

- Datasets from both synthetic and real world data
- Ranges from *2* to *960* dimensions
- The datasets are mainly on 1 million level

Approx. k-NN Graph for real Datasets

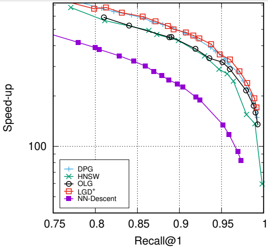


(a) Recall@1

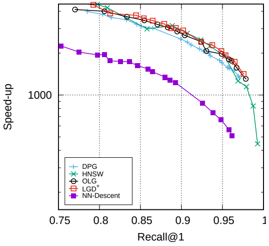
(b) Recall@10

Figure: Top-*1* and Top-*10* recall of k-NN graphs produced by NN-Descent, OLG and LGD$^+$ on eight datasets.

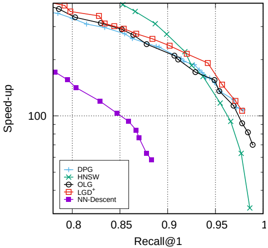
Compared to Graph-based Approach (1)



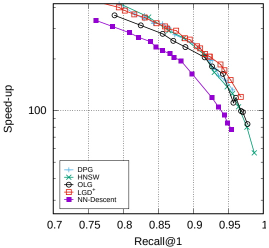
(a) SIFT1M



(b) SIFT10M

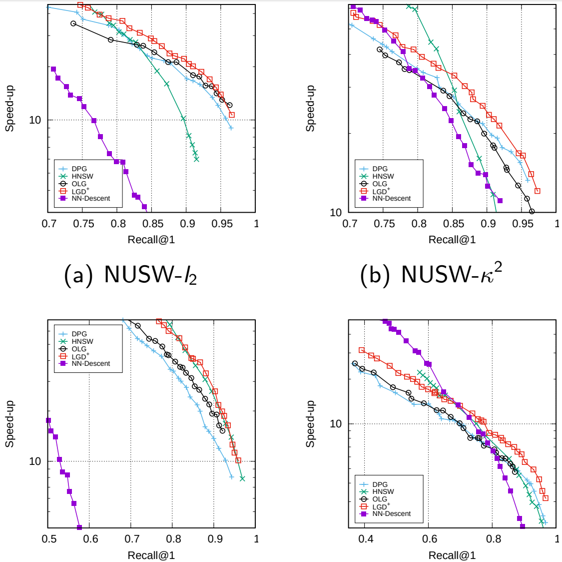


(c) GIST1M



(d) YFCC1M

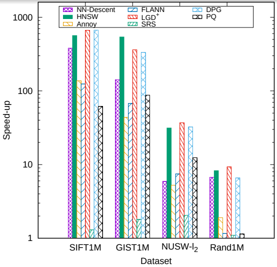
Compared to Graph-based Approaches (2)



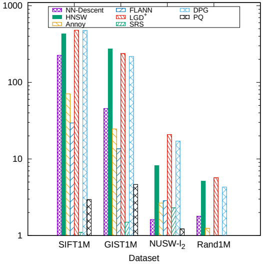
(a) NUSW-l_2 (b) NUSW-κ^2

(c) GloVe1M (d) Rand1M

Compared to state-of-the-art NN Search



(a) Recall@1=0.8



(b) Recall@1=0.9

Figure: The NN search on four datasets ranging from "easy" to "hard".

- Locality Sensitive Hashing: SRS; Tree based: FLANN
- Vector quantization: PQ;
- Graph based: ANNOY, DPG, NN-Descent, HNSW
- Best performance we could reach is related to the "intrinsic dimensions"

Summary

- Advantages
  1. NN search is performed on a dynamic graph
  2. Links to k-NNs are maintained
  3. Outperforms most of the state-of-the-art approaches
- Disadvantages
  1. Infeasible for GPU
  2. Good for query arrives one-by-one

- Publication
  1. Wan-Lei Zhao, Hui Wang, Chong-Wah Ngo, "Approximate k-NN graph construction: a generic online approach", IEEE TMM'22

Outline

1. Overview about Graph-based NN Search
2. Online Approximate k-NN Graph construction
3. k-NN Graph Merge
   - Symmetric Merge
   - Joint Merge

Outline

1. Overview about Graph-based NN Search
2. Online Approximate *k*-NN Graph construction
3. *k*-NN Graph Merge
   - Symmetric Merge
   - Joint Merge

Why Graph Merge — Scenario 1

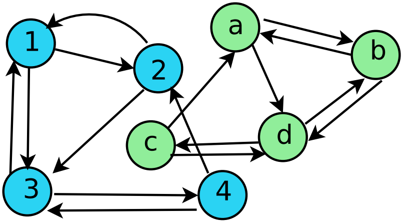


Figure: Scenario of merging two sub-graphs.

1. There are several graphs for different subsets
2. One wants to build a big graph for the whole set
3. Do not build from scratch

Why Graph Merge — Scenario 2

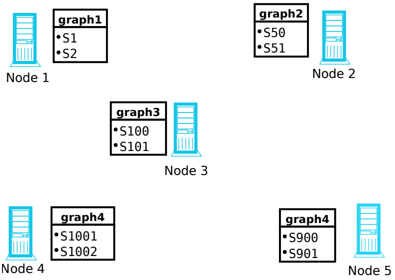


Figure: Scenario of merging two sub-graphs.

1. The dataset is big, one wants to build the graph on different nodes
2. Then merge these sub-graphs

Why Graph Merge — Scenario 3

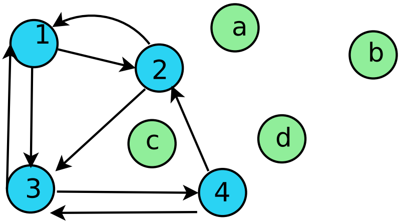
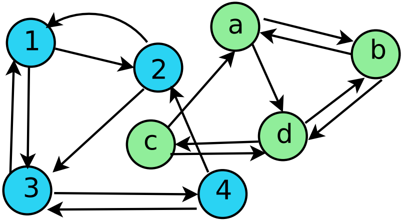


Figure: Scenario of merging two sub-graphs.

1. There are a graph
2. New samples arrive in batches

Symmetric Merge: the problem

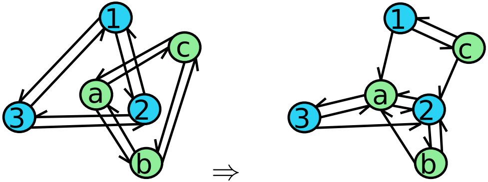


(a) Initial

Figure: Scenario of merging two sub-graphs.

- We need to combine two graphs
- We do not want to re-compute everything from scratch

Symmetric Merge: the idea (1)



(a) Initial (b) Final

Symmetric Merge: the procedure

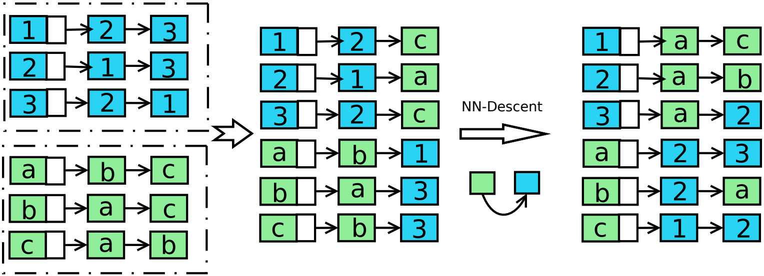


Figure: The whole flow of S-Merge.

1. Cut out rear $\frac{k}{2}$ samples from each NN list in each graph
2. Append $\frac{k}{2}$ random samples to each NN list
   - Random samples are from counter-part graph
3. Combine two grahs and perform NN-Descent
4. Merge with cut-out rear lists

Symmetric Merge (S-Merge): summary

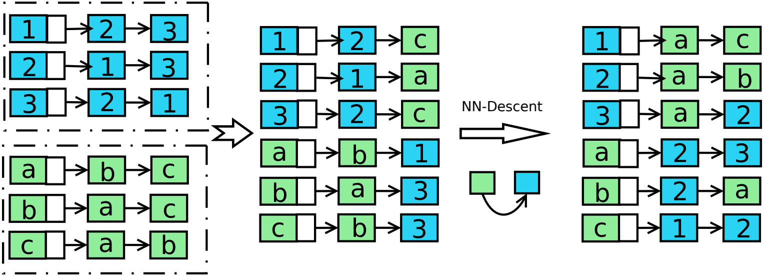


Figure: The whole flow of S-Merge.

1. Comparison happens only between samples from different graphs
2. Make use of existing sub-graph structures
3. Start from a half-baked graph, which is more cost-efffective

Outline

1. Overview about Graph-based NN Search
2. Online Approximate k-NN Graph construction
3. k-NN Graph Merge
   - Symmetric Merge
   - Joint Merge

Joint Merge: the idea

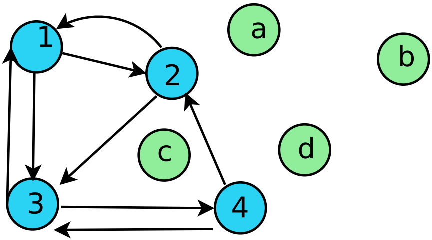
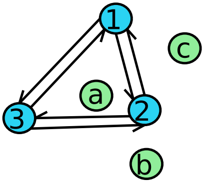


Figure: Scenario of Joint Merge.

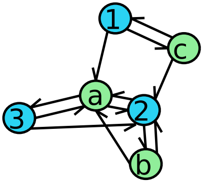
- We need to join a raw sample set into a sub-graph
- We do not want to re-compute everything from scratch
- NN-Descent + S-Merge will address this issue
- However, a better solution exists

Joint Merge: the idea



(a) Initial

$\Rightarrow$



(b) Final

- We need to join a raw sample set into a sub-graph

Joint Merge: the procedure

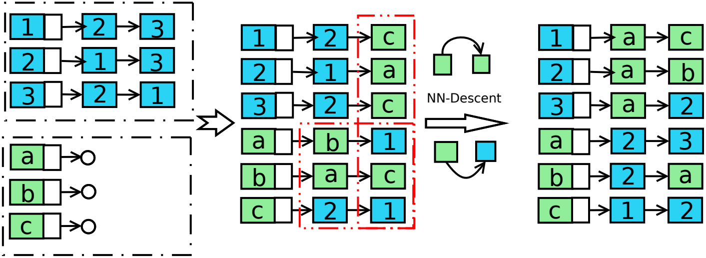


Figure: The whole flow of Joint Merge.

1. Cut out rear $\frac{k}{2}$ samples from each NN list of the 1st graph
2. Append $\frac{k}{2}$ random samples to each NN list
3. Initialize a random k-NN graph for the 2nd set
    - Random samples are from both sets
4. Combine two grahs and perform NN-Descent
5. Merge with cut-out rear lists

Joint Merge (J-Merge): summary

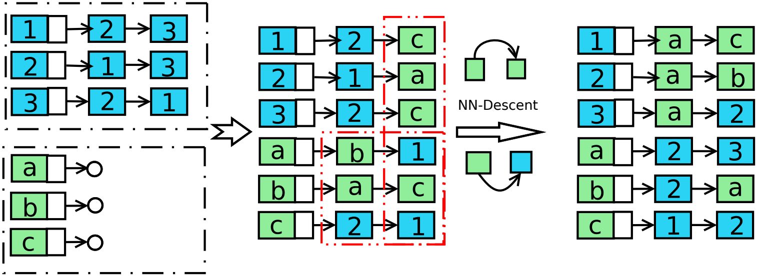


Figure: The whole flow of J-Merge.

1. Comparison happens only between samples from different graphs and within the 2nd
2. Make use of the first sub-graph structure
3. It is more efficient than NN-Descent+S-Merge solution

k-NN Graph Construction on Big data

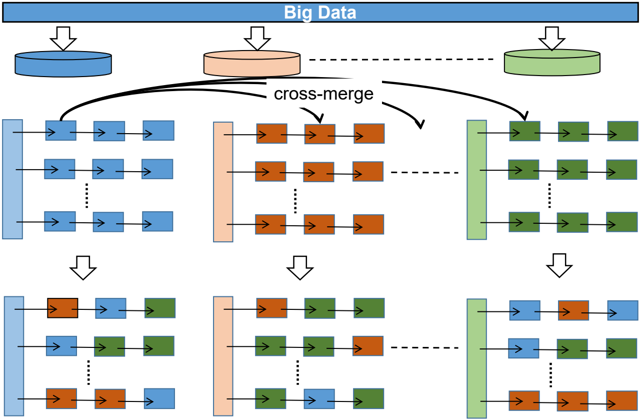
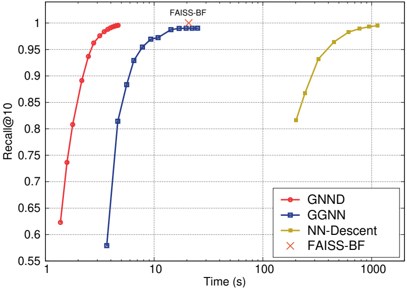


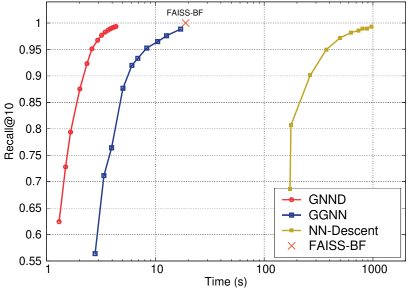
Figure: The strategy for k-NN graph construction on big data

1. Memory cannot hold the whole data, divide data into blocks
2. Build k-NN graph for each block
3. Perform cross-merging between every two sub graphs

Performance Evaluation (1)



(a) SIFT1M



(b) DEEP1M

- The merge algorithms run on NVidia 3090 GPU
- GNND is our approach

Performance Evaluation (2)

Table: Performance of k-NN graph Constr. on Billion-scale

	GNND		FAISS-IVFPQ	
Dataset	Time	*Recall@10*	Time	*Recall@10*
SIFT100M	2,583s	0.764	2,739s	0.702
SIFT100M	3,033s	0.966	4,469s	0.730
DEEP100M	2,364s	0.767	2,331s	0.705
DEEP100M	2,888s	0.956	4,262s	0.770
SIFT1B	77h	0.955	-	-
DEEP1B	76h	0.951	-	-

- Outperforms state-of-the-art by large margin
- We are able to construct k-NN graph for billion-scale data with high quality

S-Merge and J-Merge: summary

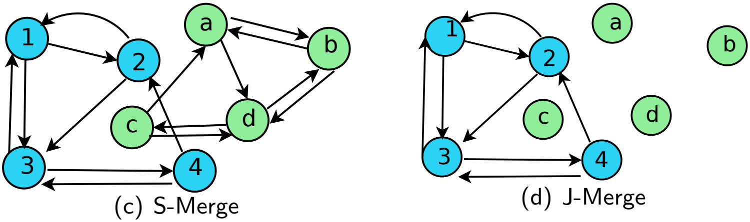
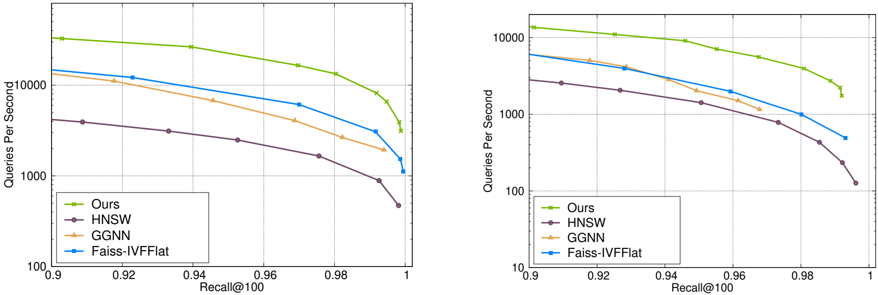


Figure: Two types of *k*-NN graph merge.

- This approach has been integrated in a face-recogition system
- Serves for more than 6 million people
- Publications
  1. Wan-Lei Zhao, Hui Wang, Peng-Cheng Lin, Chong-Wah Ngo, "On the Merge of k-NN graph", IEEE TBD'21
  2. Hui Wang, Wan-Lei Zhao, et.al, "Fast k-NN Graph Construction by GPU based NN-Descent", CIKM'21

Recent Progress: work submitted



(a) SIFT10M (b) T2I10M

Figure: NN search on GPU.

- Experiments are pulled out on NVidia 3090
- Outperforms all the approaches in the literature
- $> 30,000$ queries per second

Q & A

Thanks for your attention!