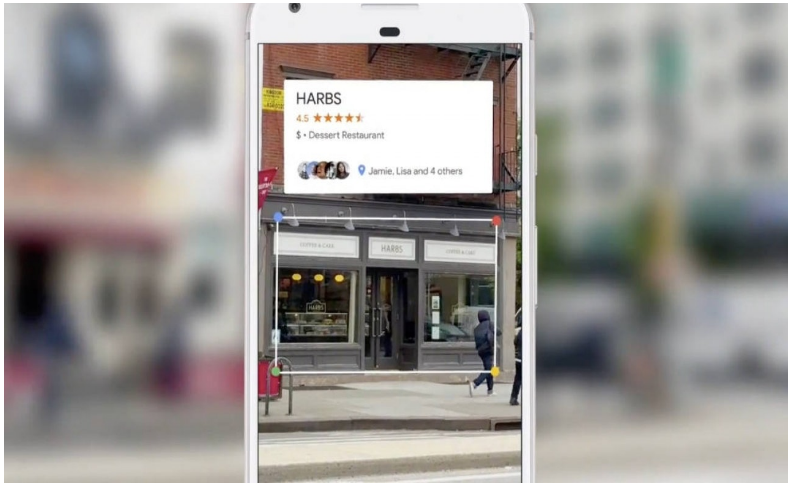
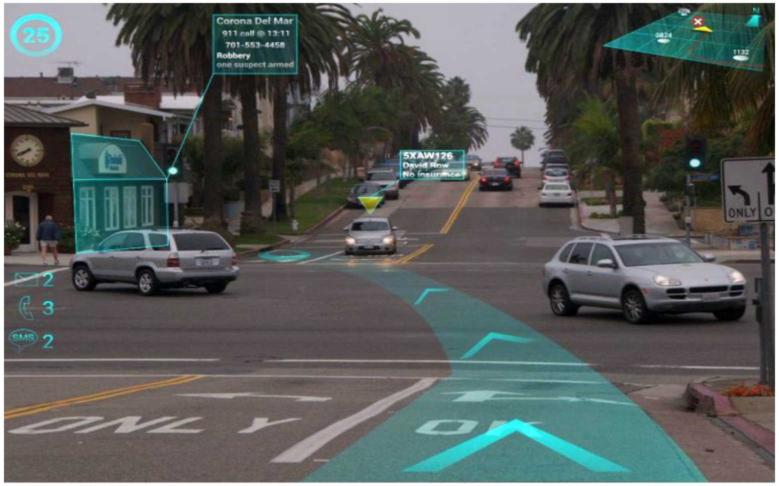
FoggyCache: Cross-Device Approximate Computation Reuse

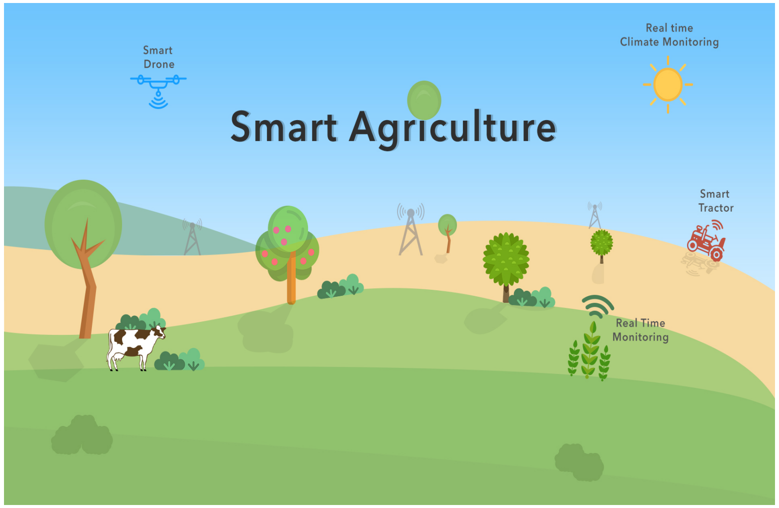
Peizhen Guo, Bo Hu, Rui Li, Wenjun Hu

Yale University

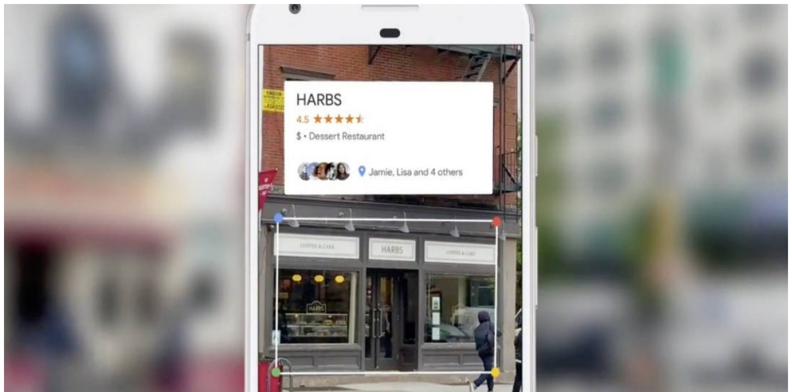
Emerging trend



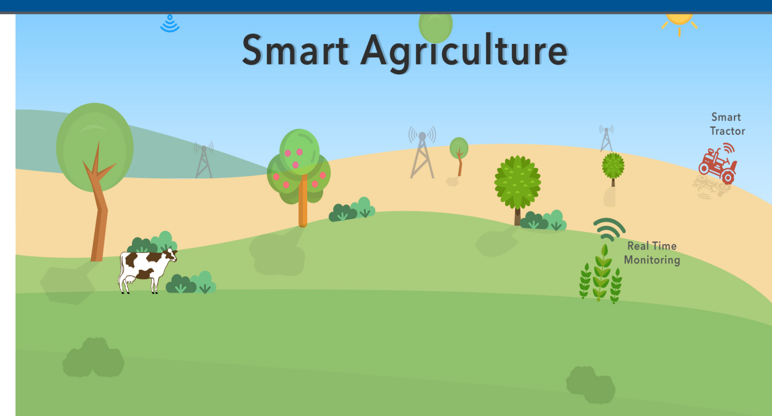




Emerging trend

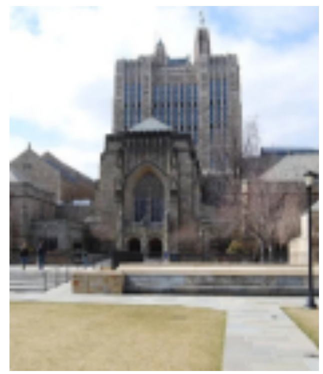
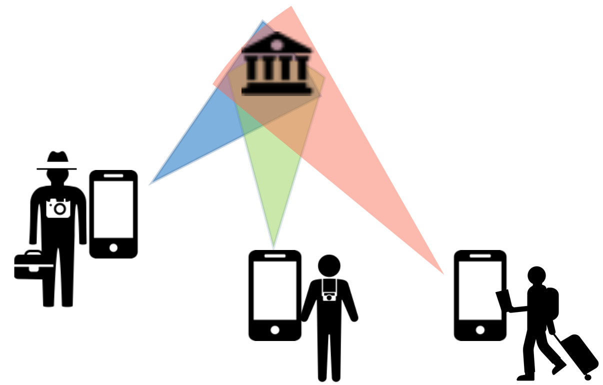


Computation intensive:
incurring offloading latency, draining battery



Same, popular apps run on nearby devices

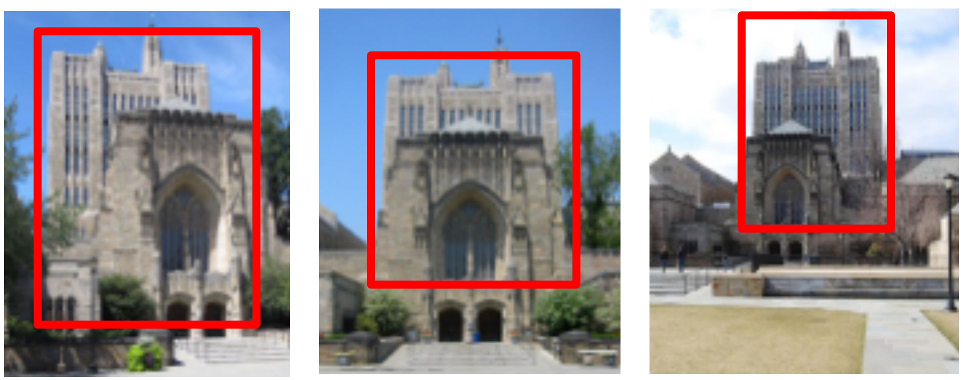


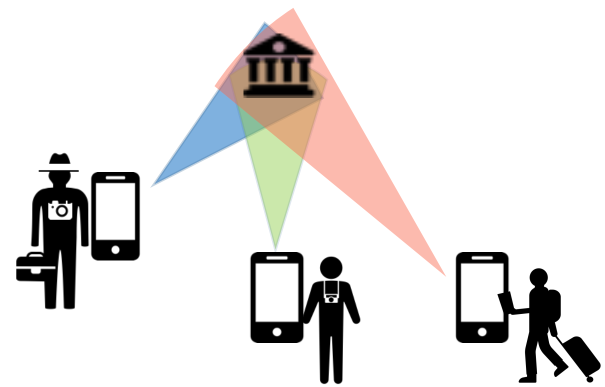



Example: landmark recognition

Redundancy across nearby devices

“Sterling Library”

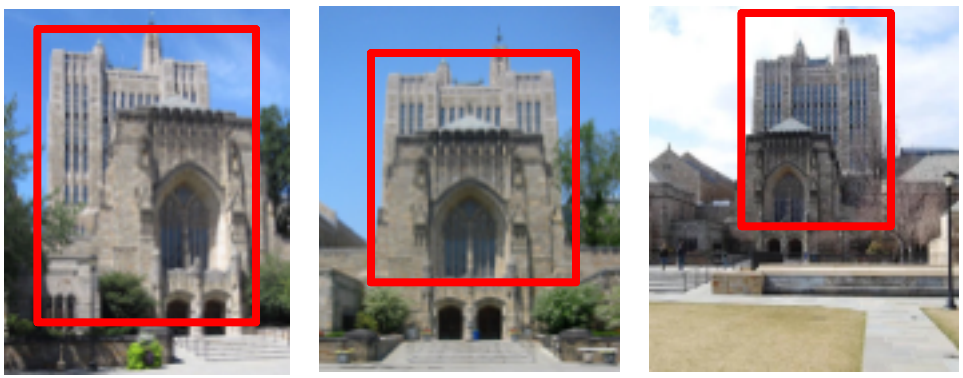


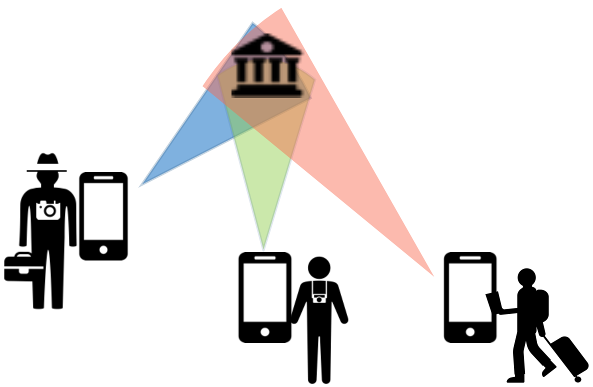


Example: landmark recognition

Redundancy across nearby devices

“Sterling Library”

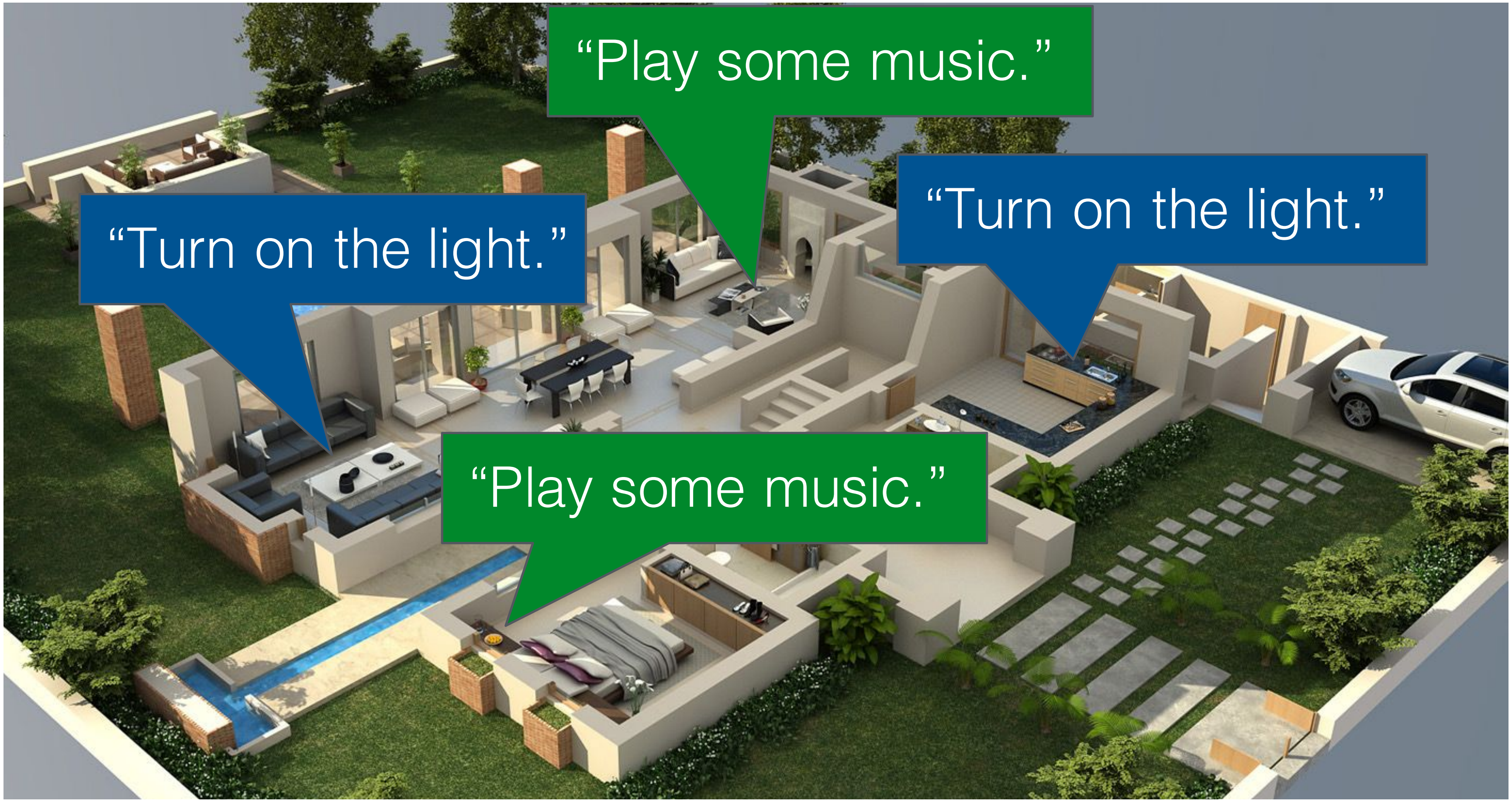




Example: landmark recognition

Up to 82% input generate the same result

More Examples: smart home scenarios

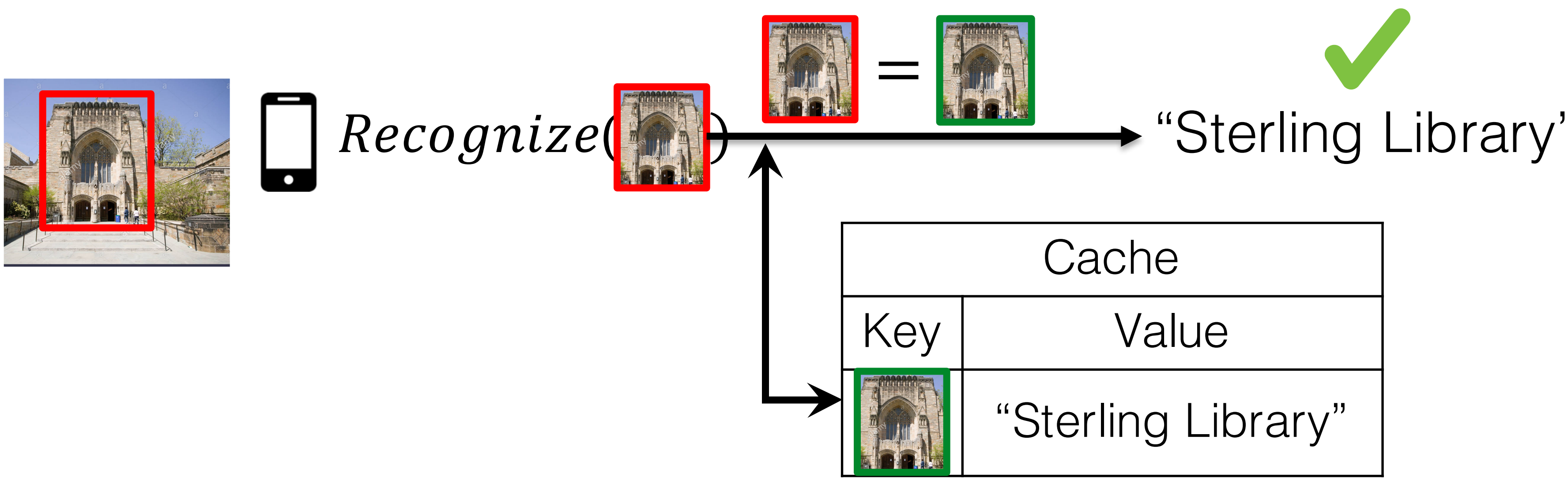


Can we eliminate this redundancy?

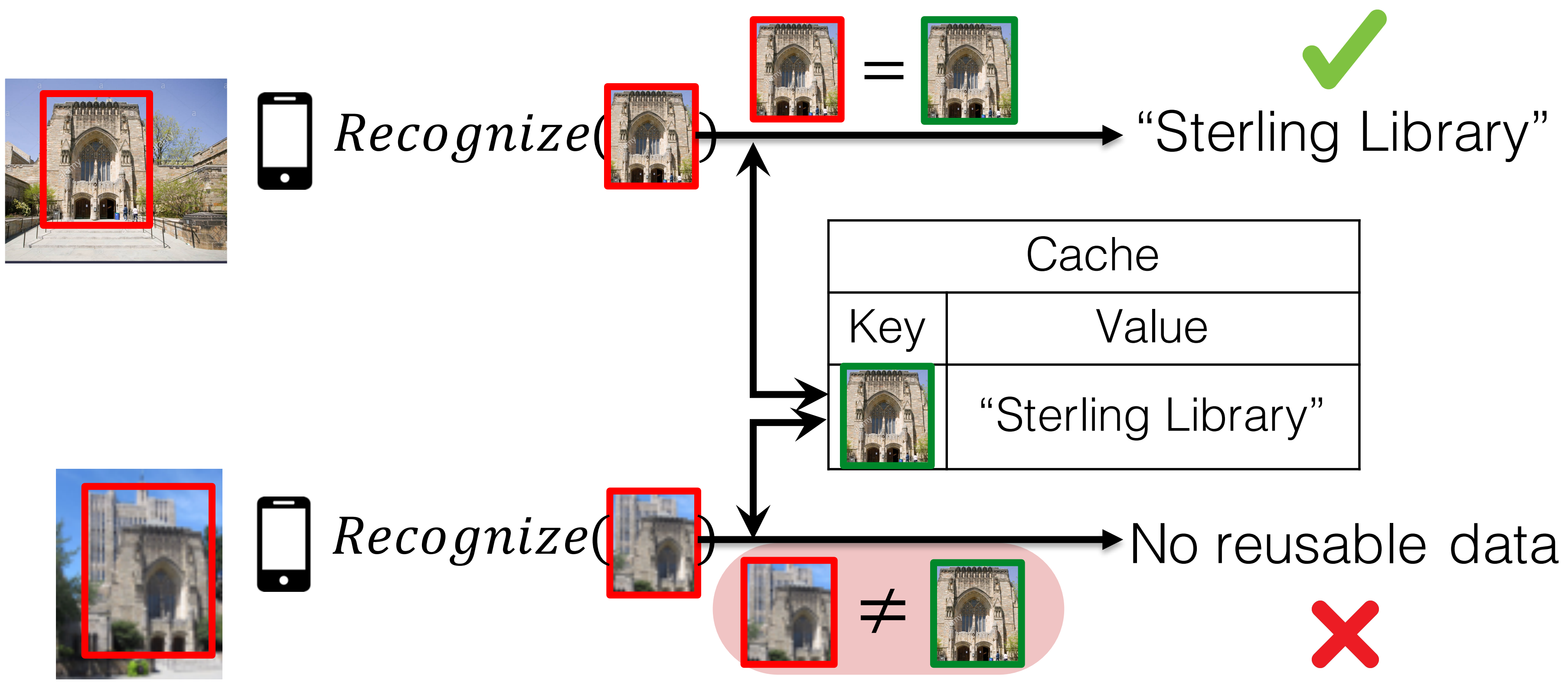
Can we eliminate this redundancy?

Reuse previous computation results

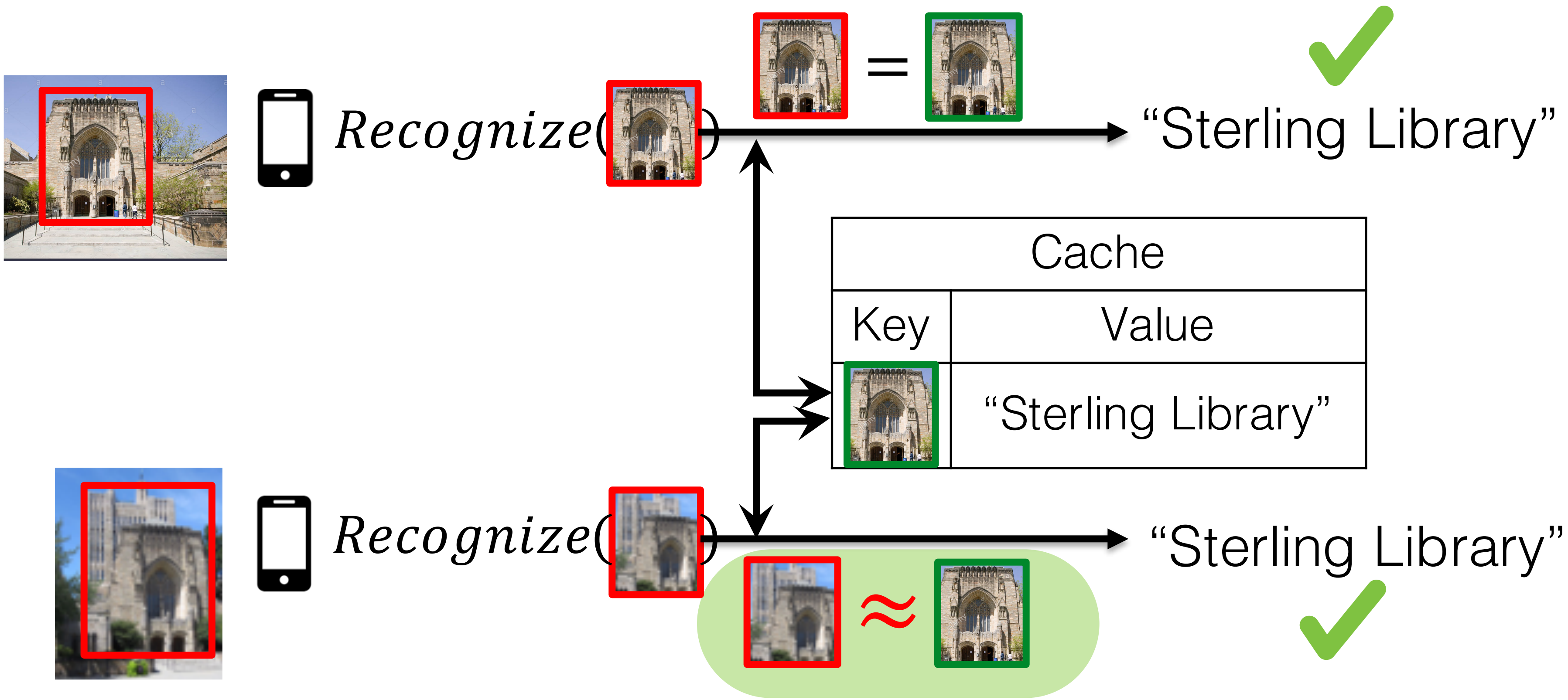
Traditional computation reuse



Traditional computation reuse



Ideal computation reuse



Approximate
Computation Reuse

Our goals

- Algorithms for *approximate computation reuse*

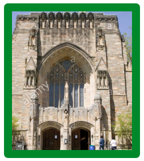
- A system to eliminate redundancy across devices

Reuse process

Reuse process

Input data

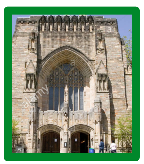
Lookup computation records with similar input

[ , "Bass Library"] [ , "Sterling Library"] ……

Reuse process

Input data

Lookup computation records with similar input

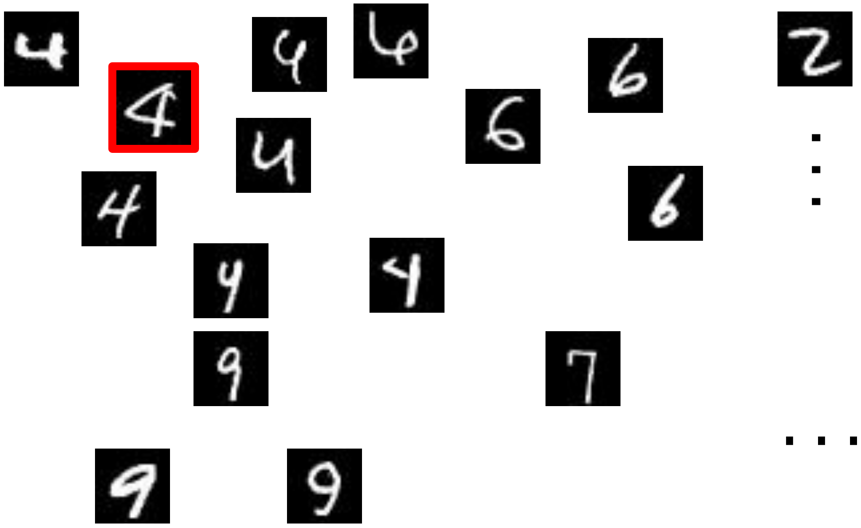
[ , “Bass Library”]   [ , “Sterling Library”] . . . . . .

Determine reuse outcome

Not reusable or *“Bass Library”* or *“Sterling Library”*

The rest of the talk…

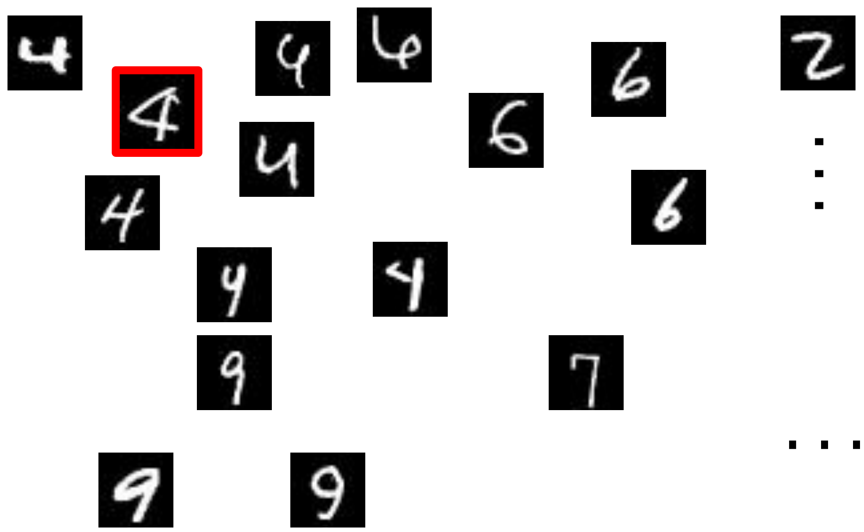
- Algorithms for *approximate computation reuse*
    - *A-LSH* – fast lookup
    - *H-kNN* – reuse with accuracy guarantee
- FoggyCache system for cross-device reuse



Handwritten digits from MNIST dataset

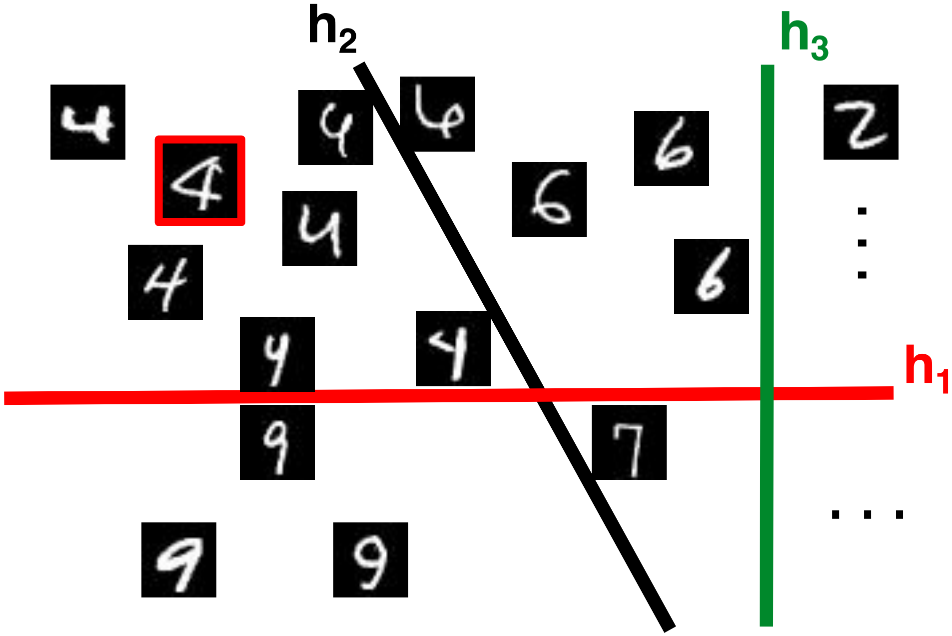
A-LSH: strawman

Locality sensitive Hashing (LSH)



A-LSH: strawman

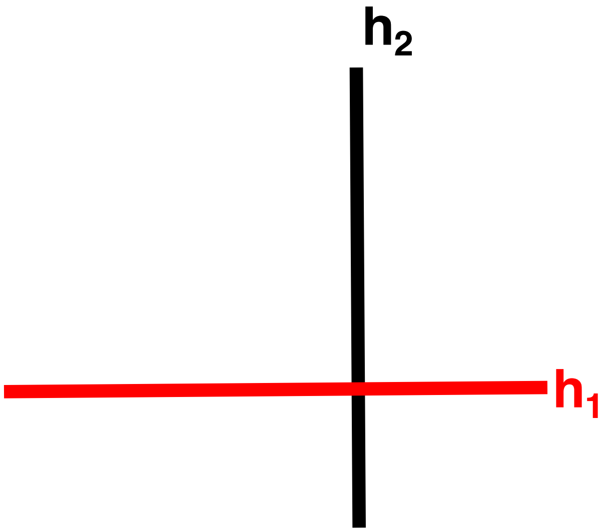
Locality sensitive Hashing (LSH)

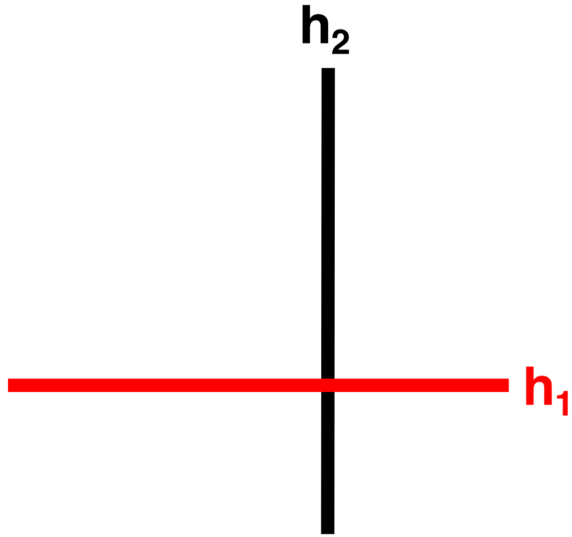


More similar data stay in the same bucket with higher probability

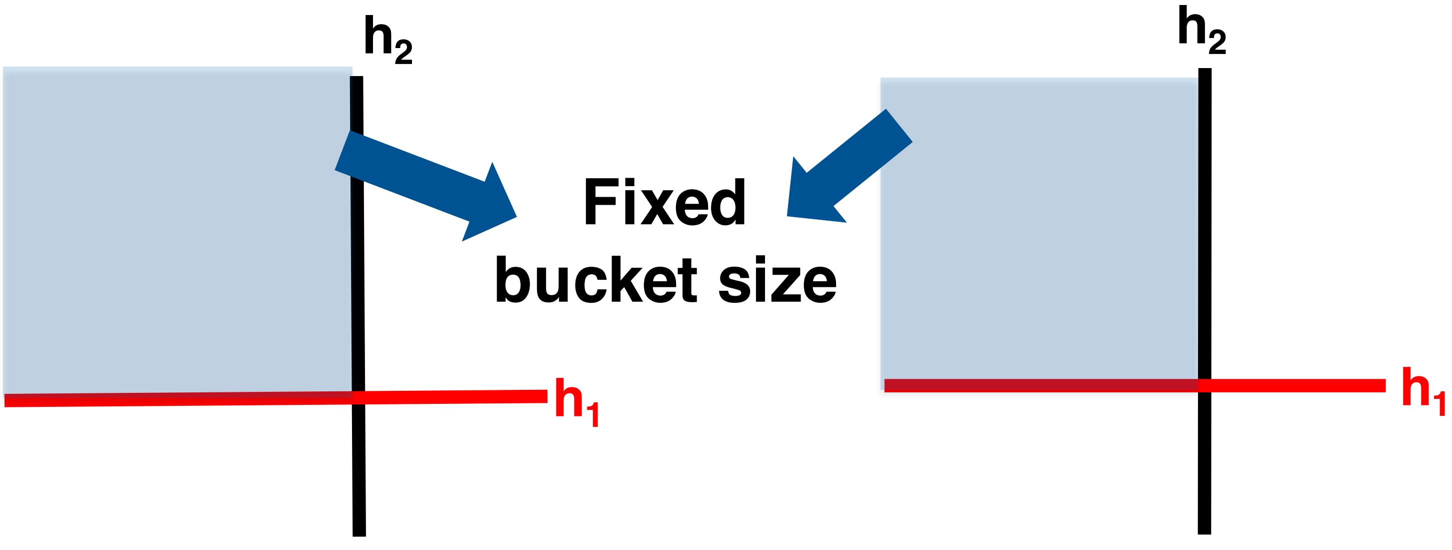
LSH is not enough

LSH is not enough

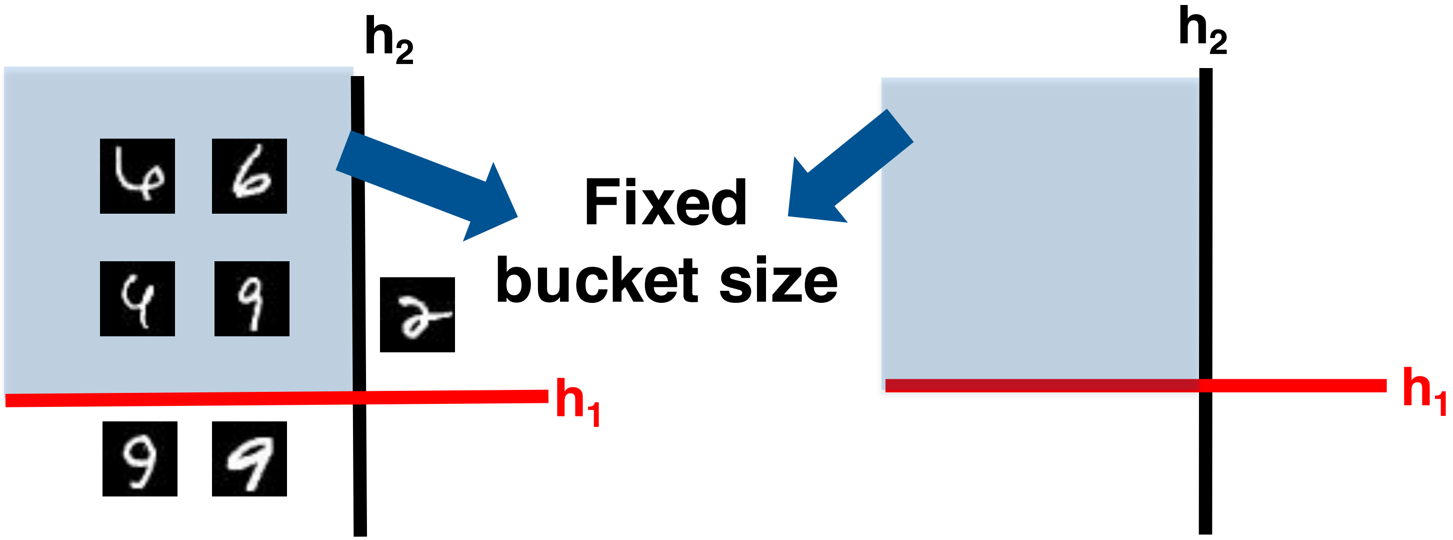




LSH is not enough

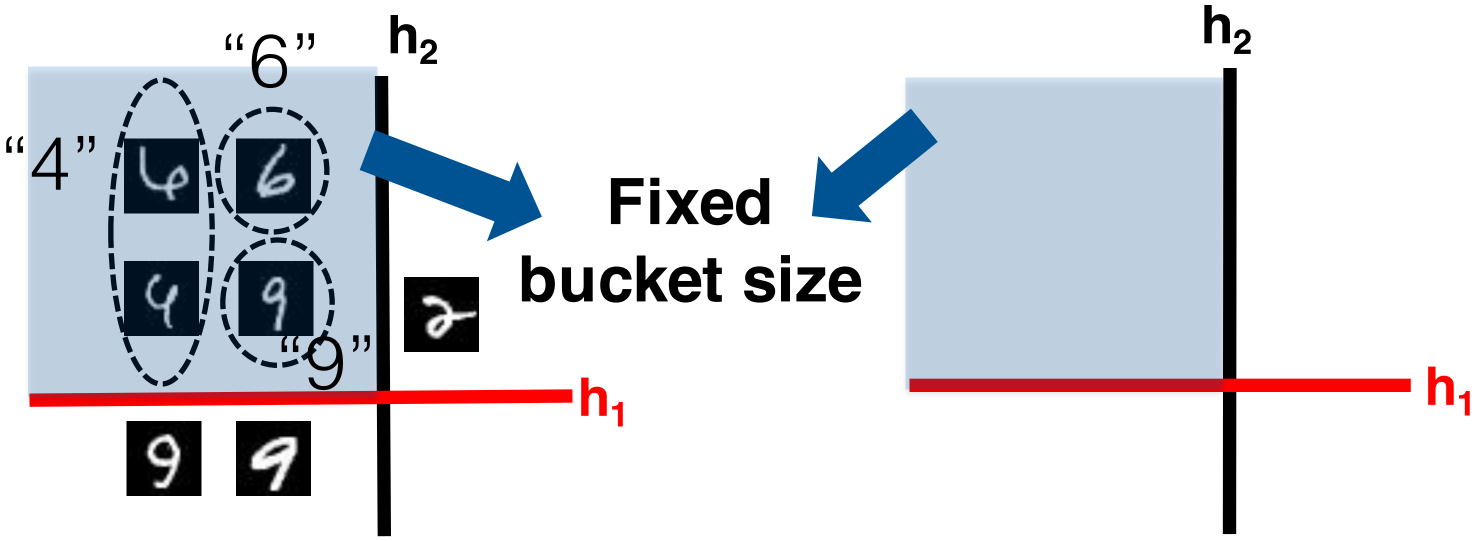


LSH is not enough



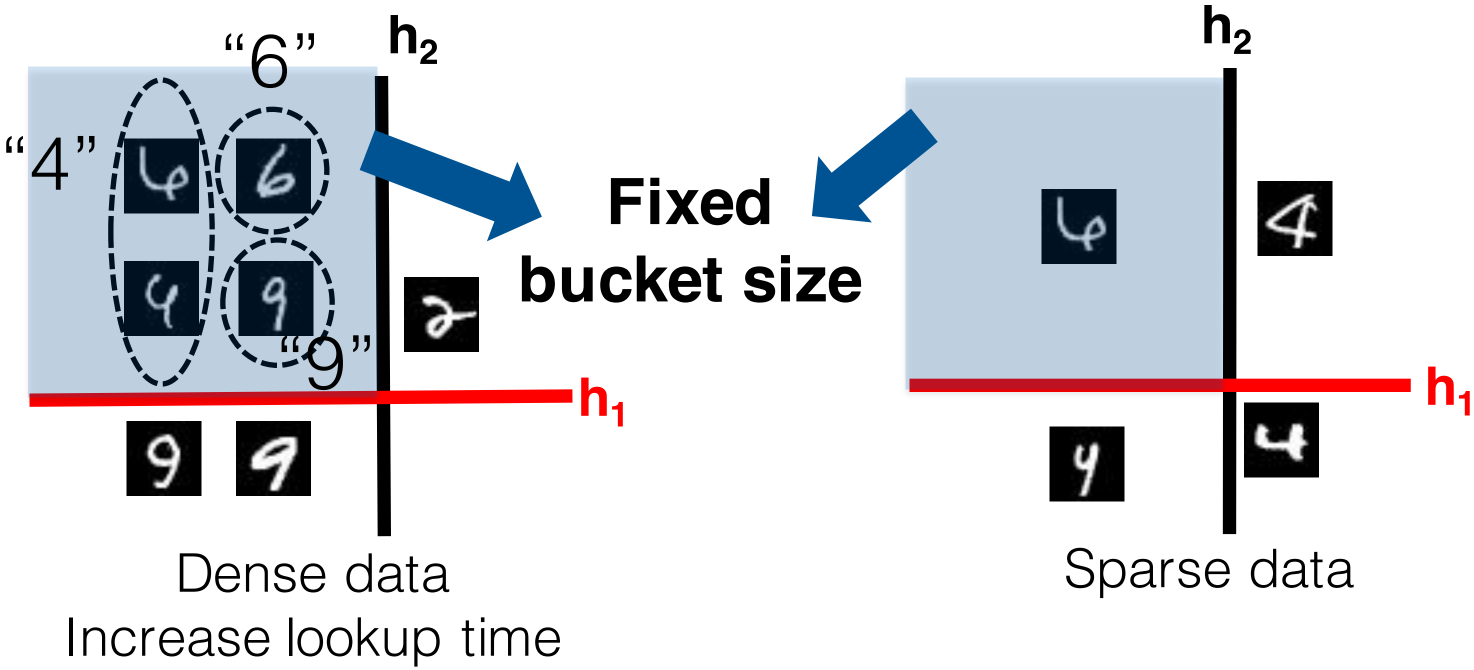
Dense data

LSH is not enough

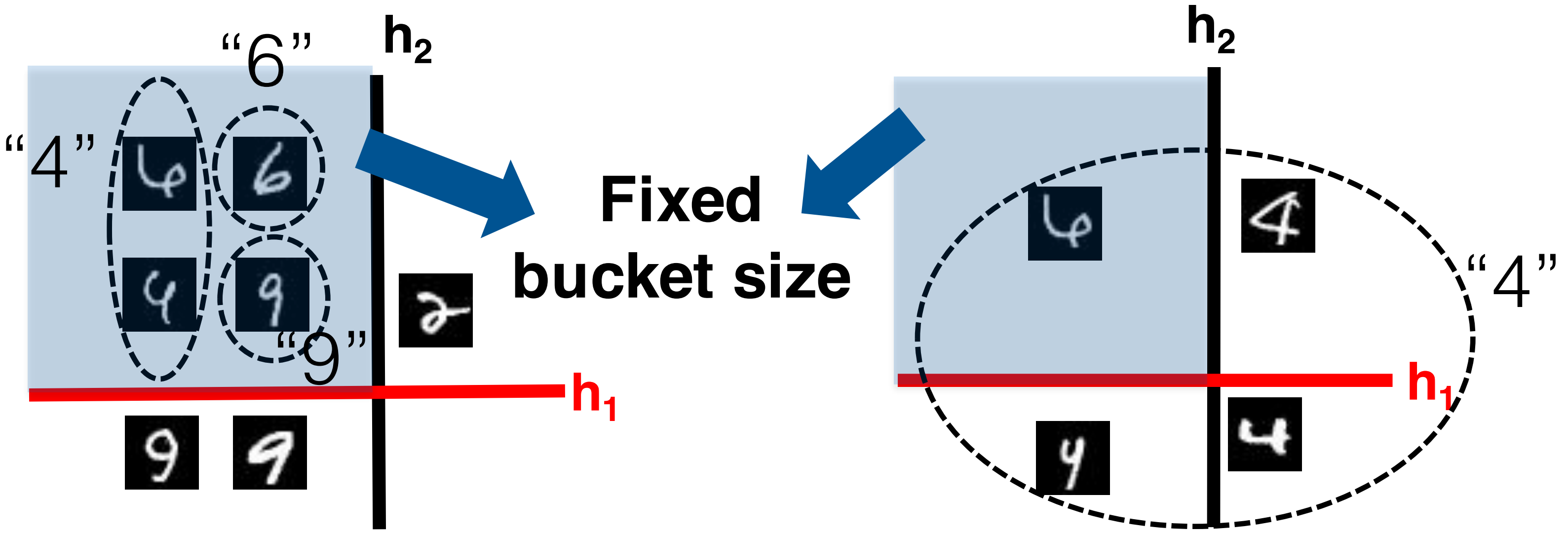


Dense data
Increase lookup time

LSH is not enough



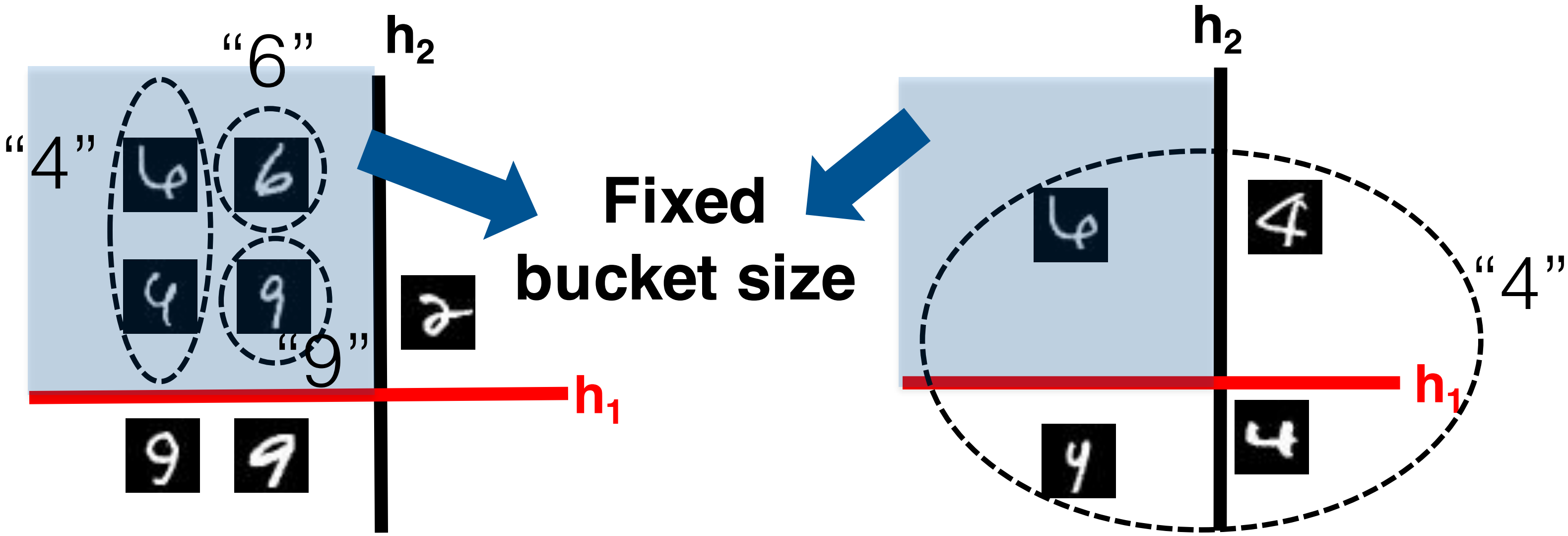
LSH is not enough



Dense data
Increase lookup time

Sparse data
Miss actually similar records

LSH is not enough



Dense data
Increase lookup time

Sparse data
Miss actually similar records

LSH configuration is static

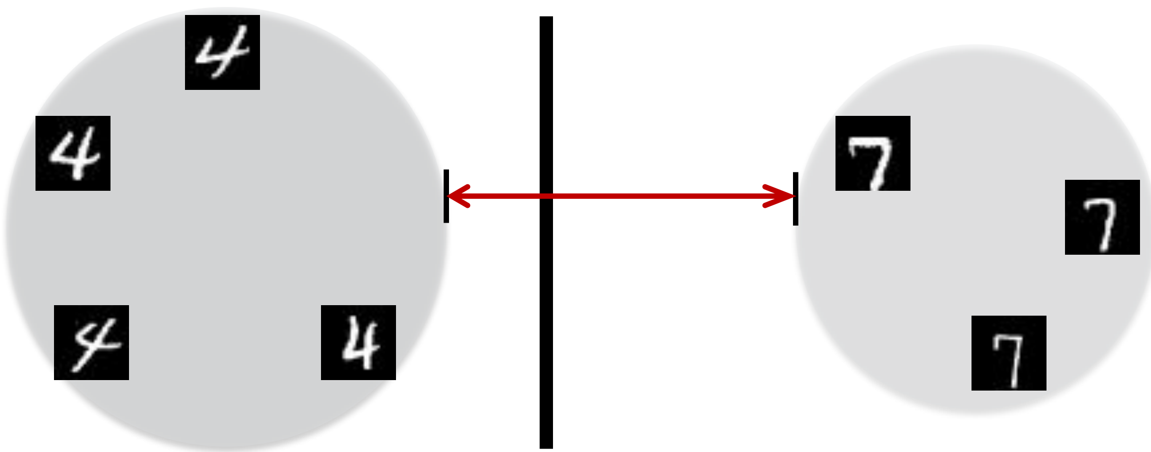
Data distribution is dynamic

Adaptive-LSH

adapt the bucket size to data distribution

Adaptive-LSH

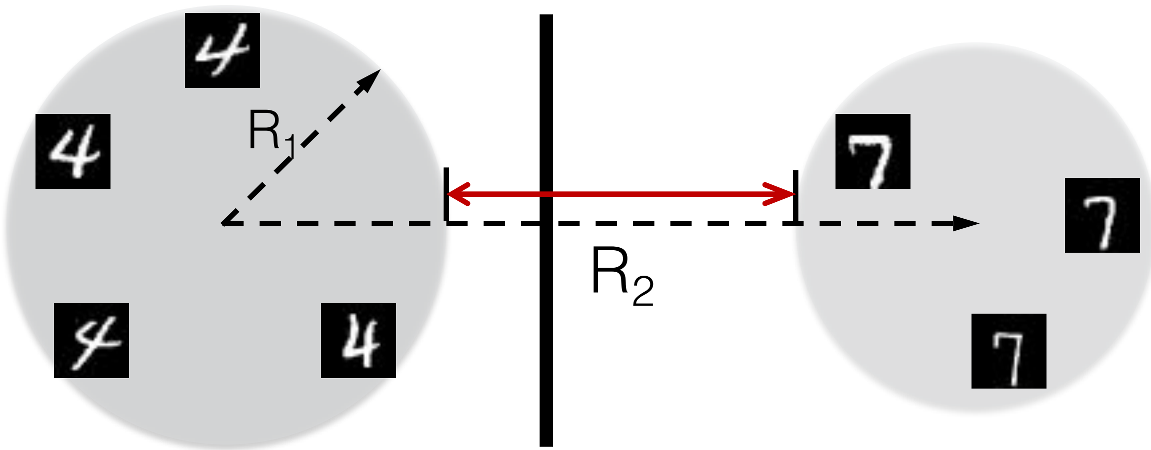
adapt the bucket size to data distribution



Proper bucket setting

Adaptive-LSH

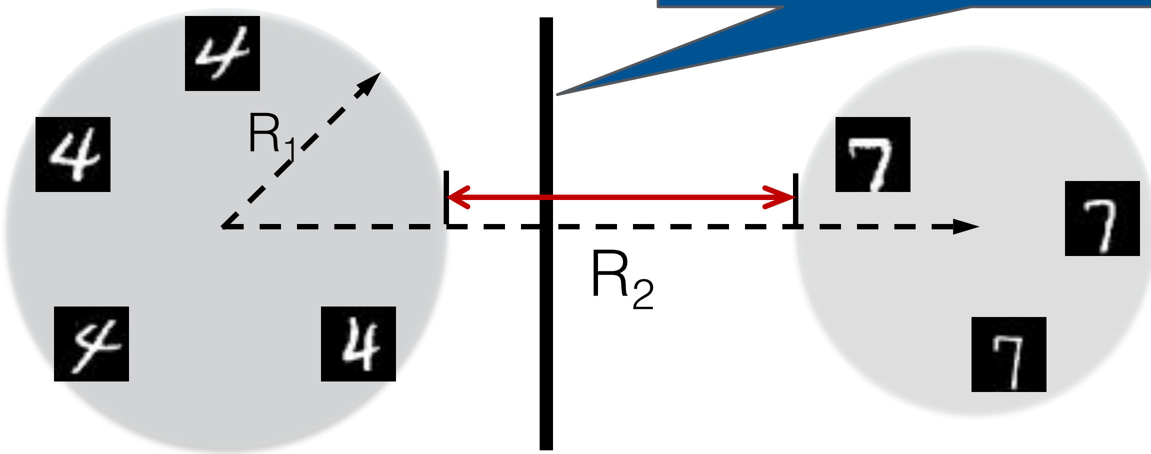
adapt the bucket size to data distribution



Step 1: Use the ratio ***c=R2/R1*** to characterize input data distribution

Adaptive-LSH

Recall: the bucket size also affects lookup time

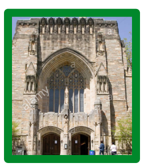


Step 2: Adapt bucket size according to ***c*** and the lookup time target

Reuse process


Input data

Lookup computation records with similar input

[ , "Bass Library"] [ , "Sterling Library"] ·······

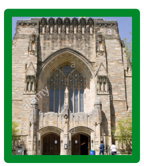
Determine reuse outcome

Not reusable or *"Bass Library"* or *"Sterling Library"*

Reuse process

Input data

Lookup computation records
with similar input

[ , “Bass Library”] [ , “Sterling Library”] ……

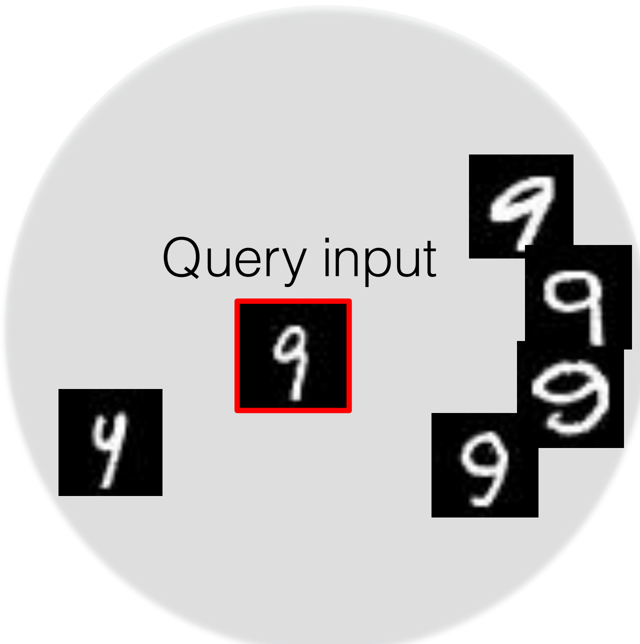
Determine reuse outcome

Not reusable or *“Bass Library”* or *“Sterling Library”*

H-kNN: strawman

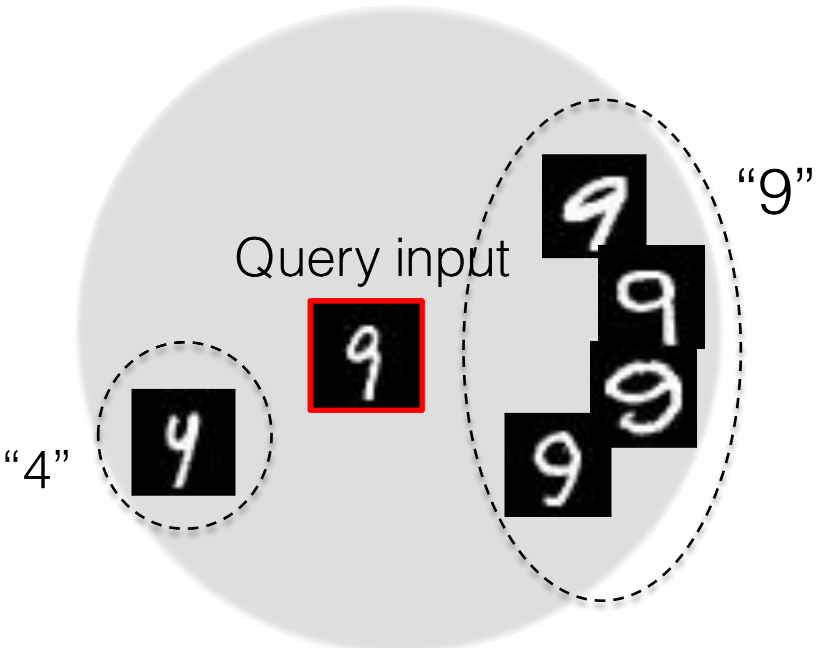
Basic idea

k Nearest Neighbor



H-kNN: strawman

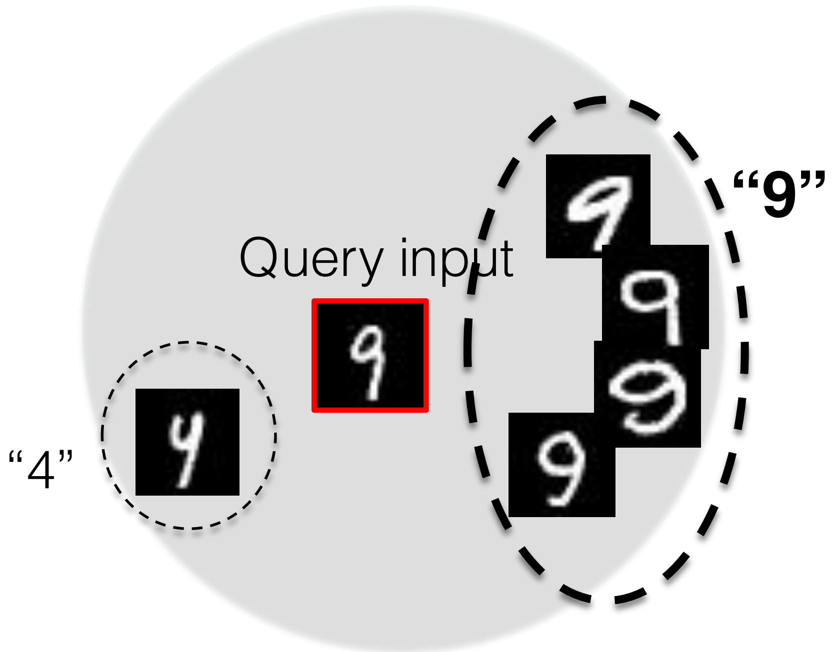
Basic idea
k Nearest Neighbor



H-kNN: strawman

Basic idea

k Nearest Neighbor



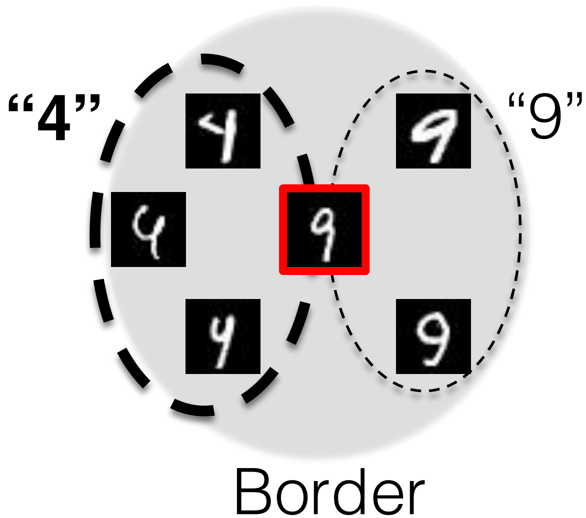
Take the result label of the largest cluster as the reuse outcome

kNN not enough

kNN not enough

Label of the largest cluster is not always the desirable reuse result

kNN not enough



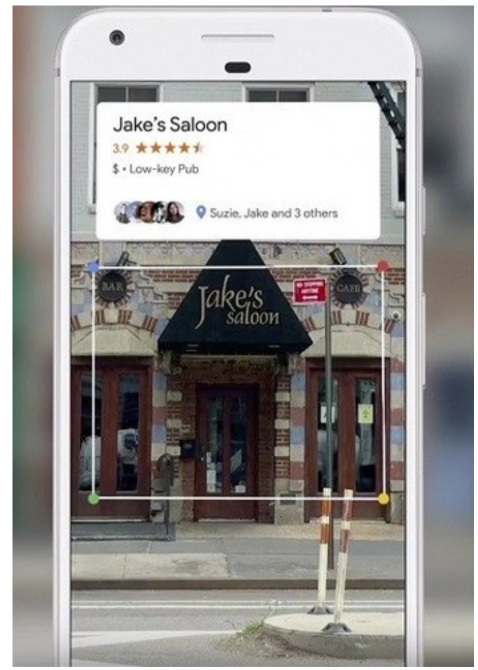
Label of the largest cluster is not always the desirable reuse result

kNN not enough

Need high accuracy

Prefer less computation

Pill recognition

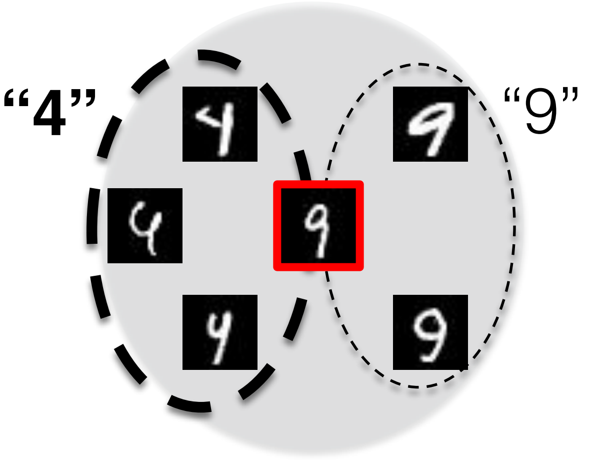


Google Lens

kNN does not give us control over the trade-off

kNN: what is needed?

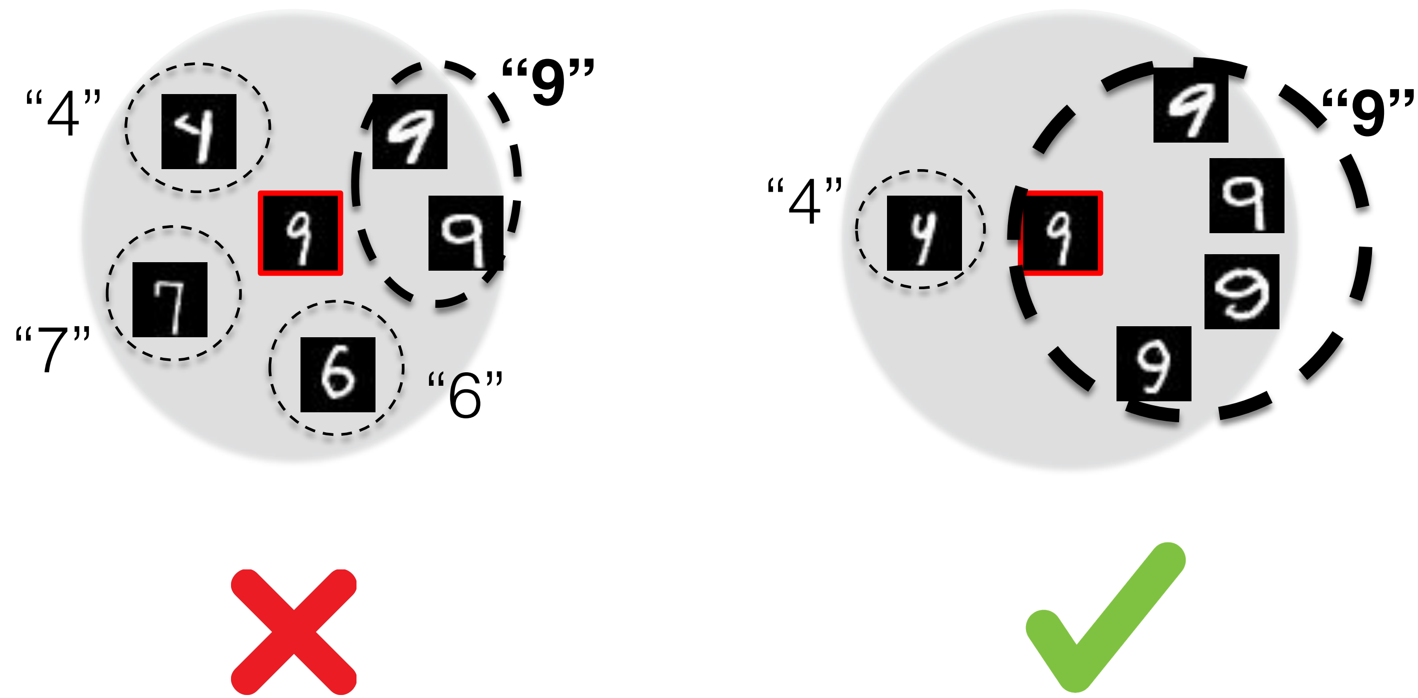
kNN: what is needed?



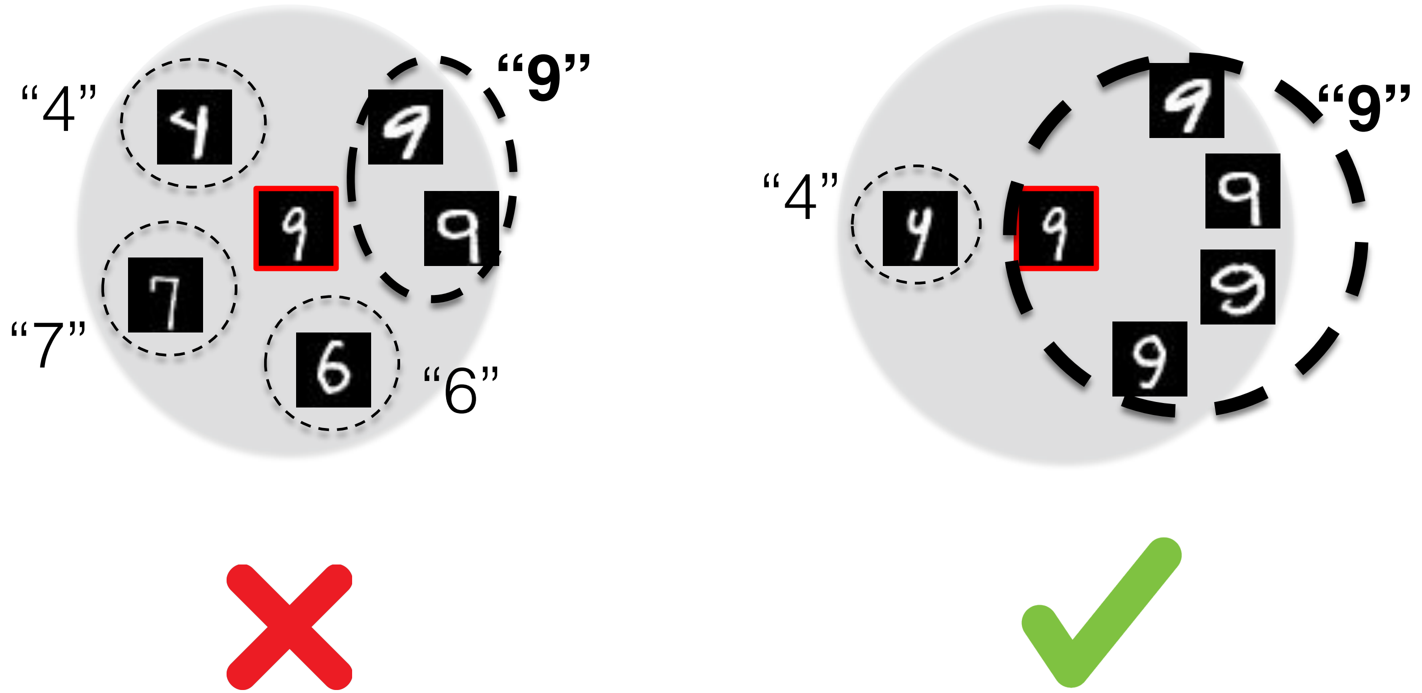
Need to gauge dominance level of clusters

Why dominance level matters?

Why dominance level matters?

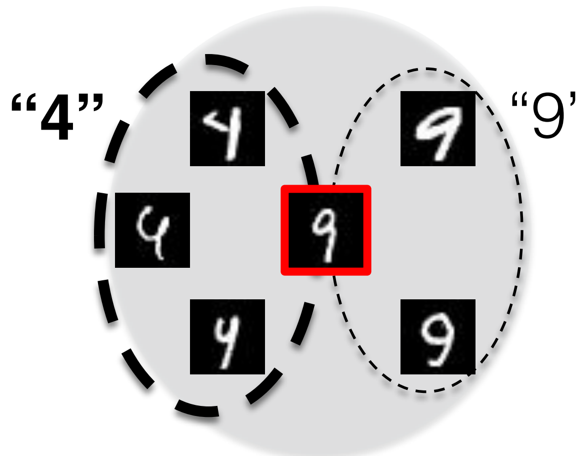


Why dominance level matters?



A more dominant cluster
→ more confidence of accurate reuse

kNN: what is needed?

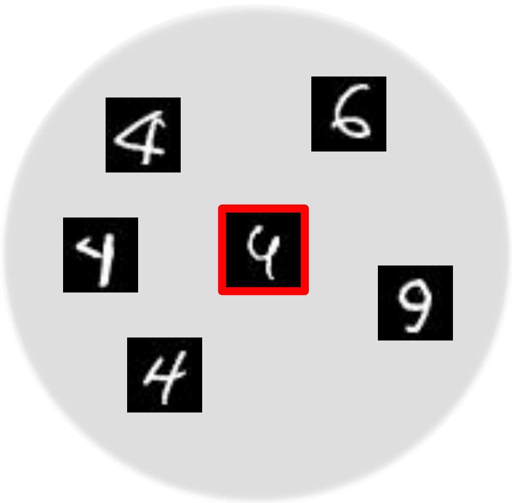


Need to gauge dominance level of clusters

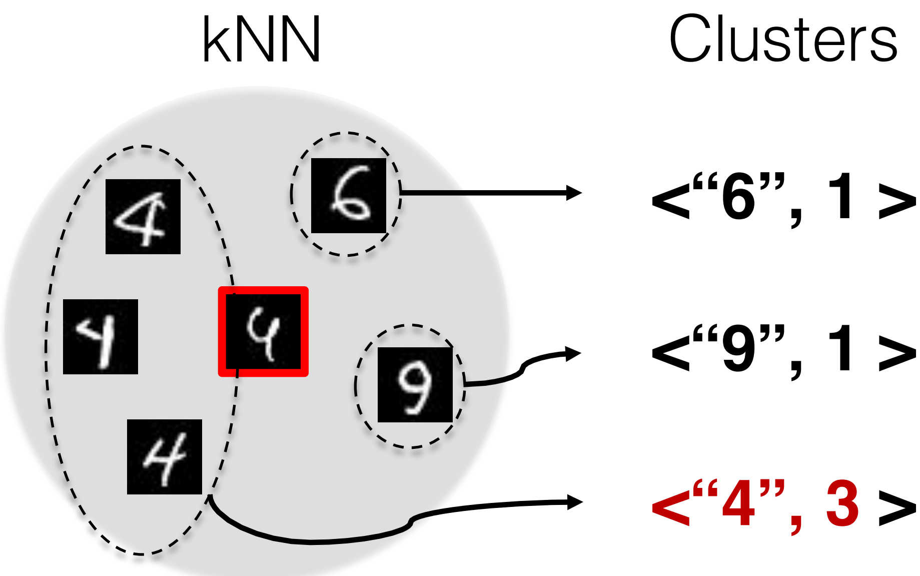
Can then customize reuse trade-off

Homogeneity factor

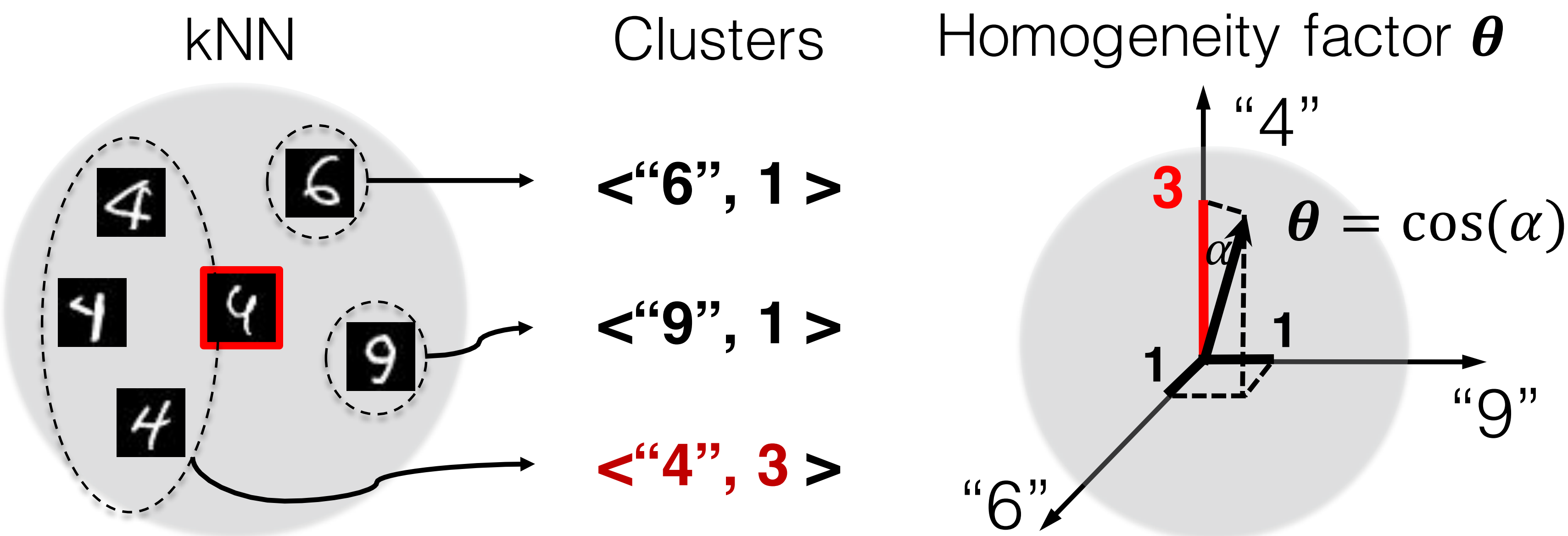
kNN



Homogeneity factor

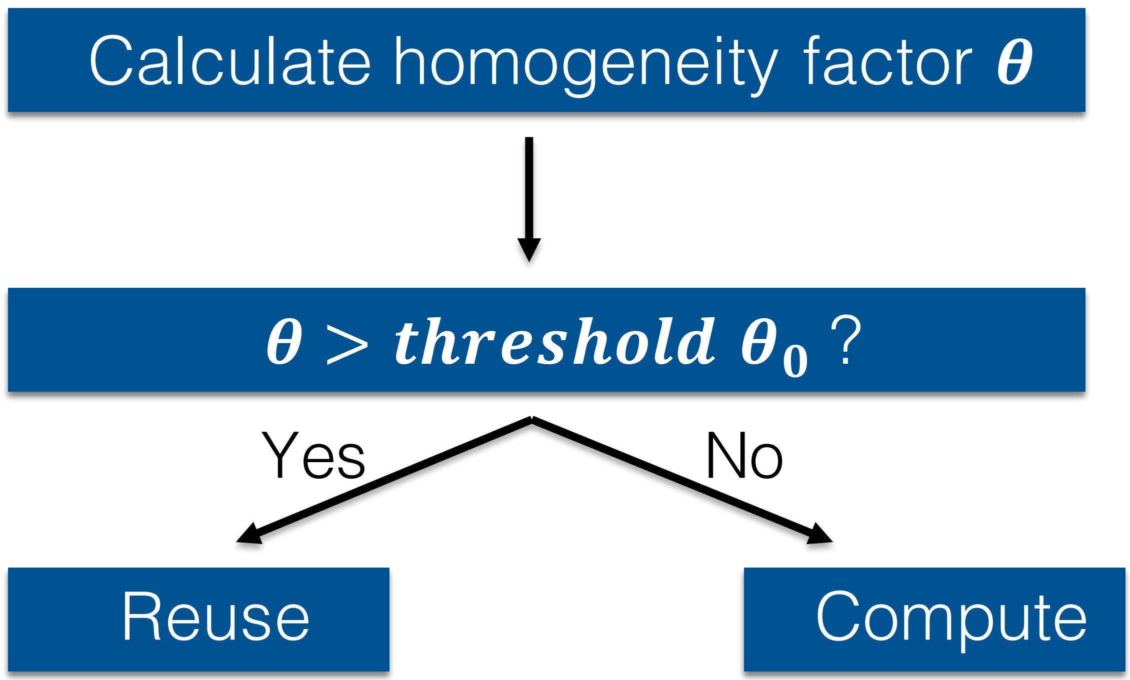


Homogeneity factor



A high $\boldsymbol{\theta}$ ⇒ a large dominant cluster label (i.e., "4")
⇒ a high confidence of correct reuse.

Homogemized-kNN (H-kNN)

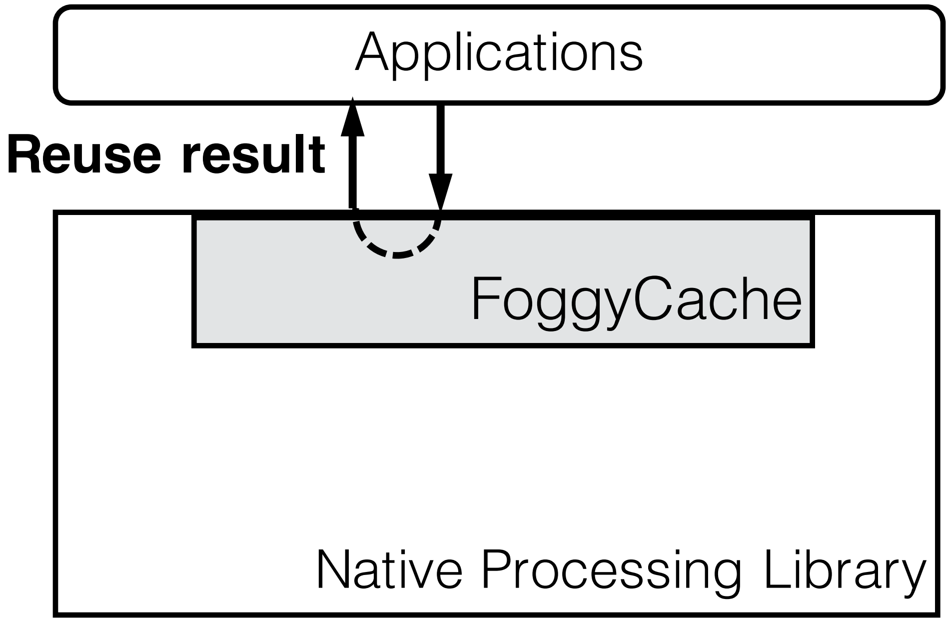


Approximate computation reuse

- Algorithms for *approximate computation reuse*
  - *A-LSH* – fast lookup
  - *H-kNN* – reuse with accuracy guarantee
- **FoggyCache system for cross-device reuse**

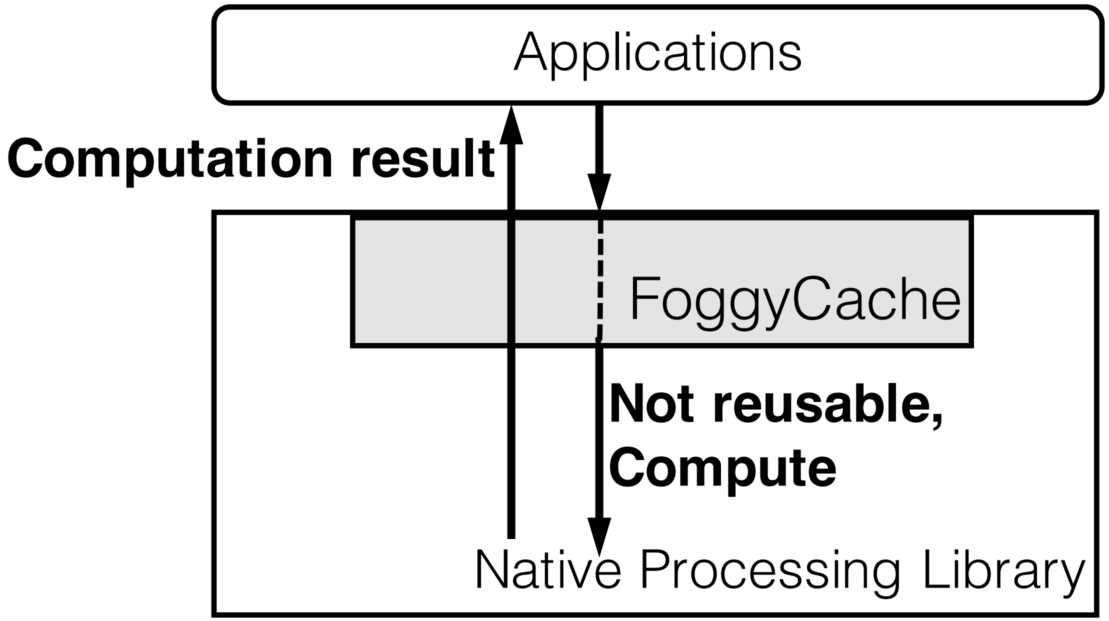
FoggyCache architecture

- FoggyCache intercepts at library level



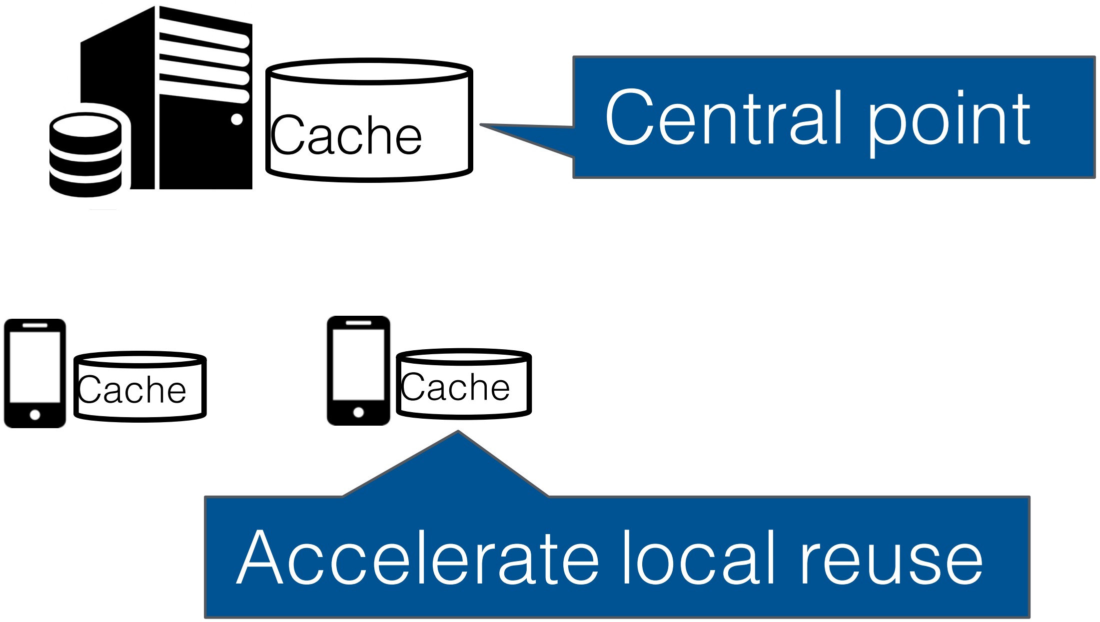
FoggyCache architecture

- FoggyCache intercepts at library level

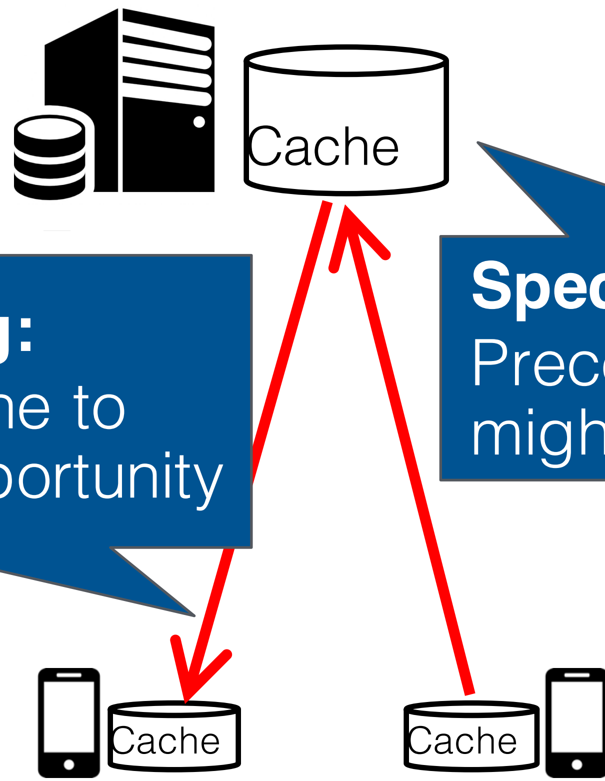


FoggyCache architecture

- Cache is deployed at both edge server and client



System optimizations



Stratified sampling:
Populate client cache to maximize reuse opportunity

Speculative execution:
Precompute results that might be reused.

Details in the paper

Performance

General setup

Devices


Linux desktop


Google Nexus 9

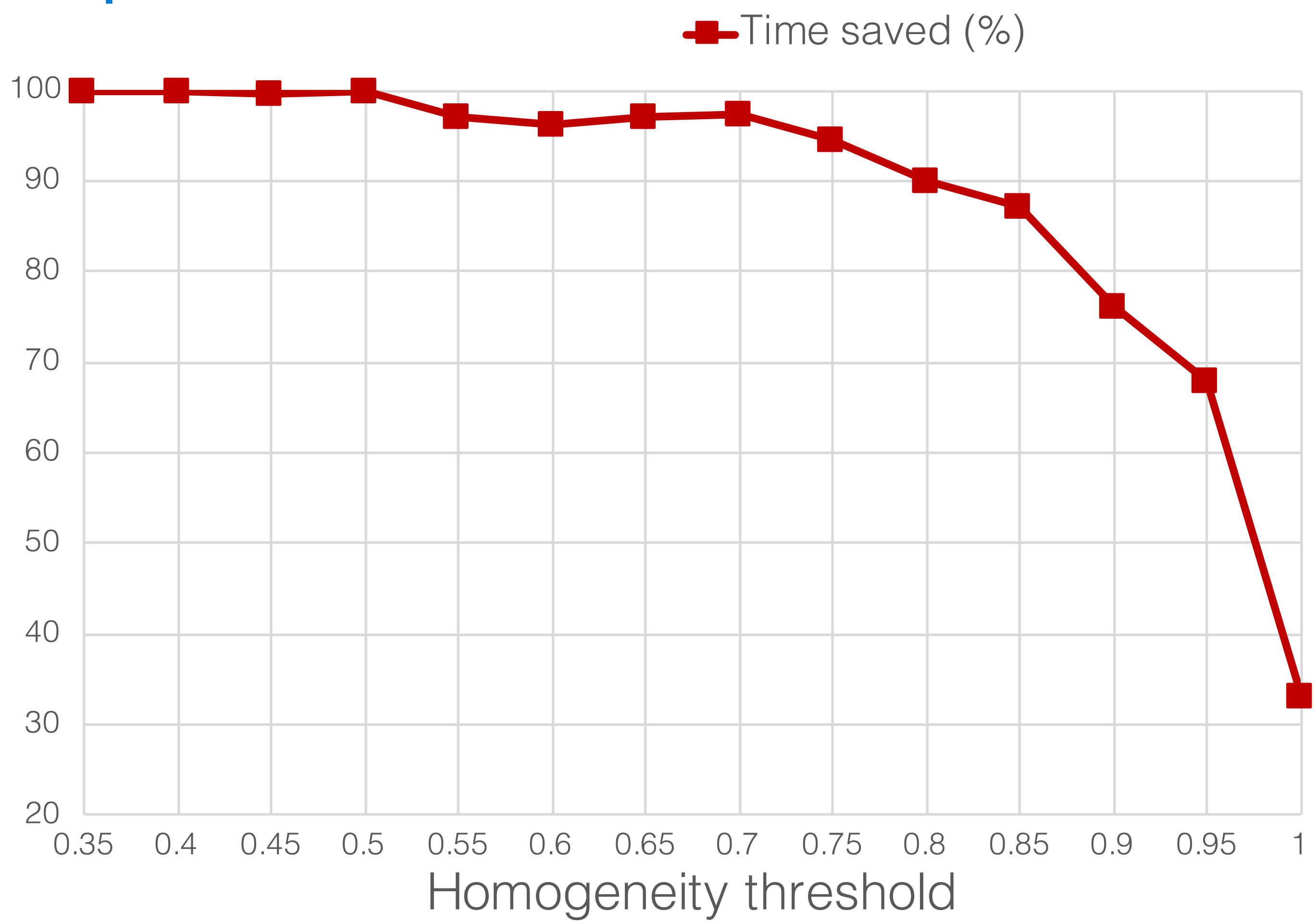
Visual workloads & datasets

- Plant recognition: *ImageNet* subset
- Landmark recognition: *Oxford Buildings*, video feeds

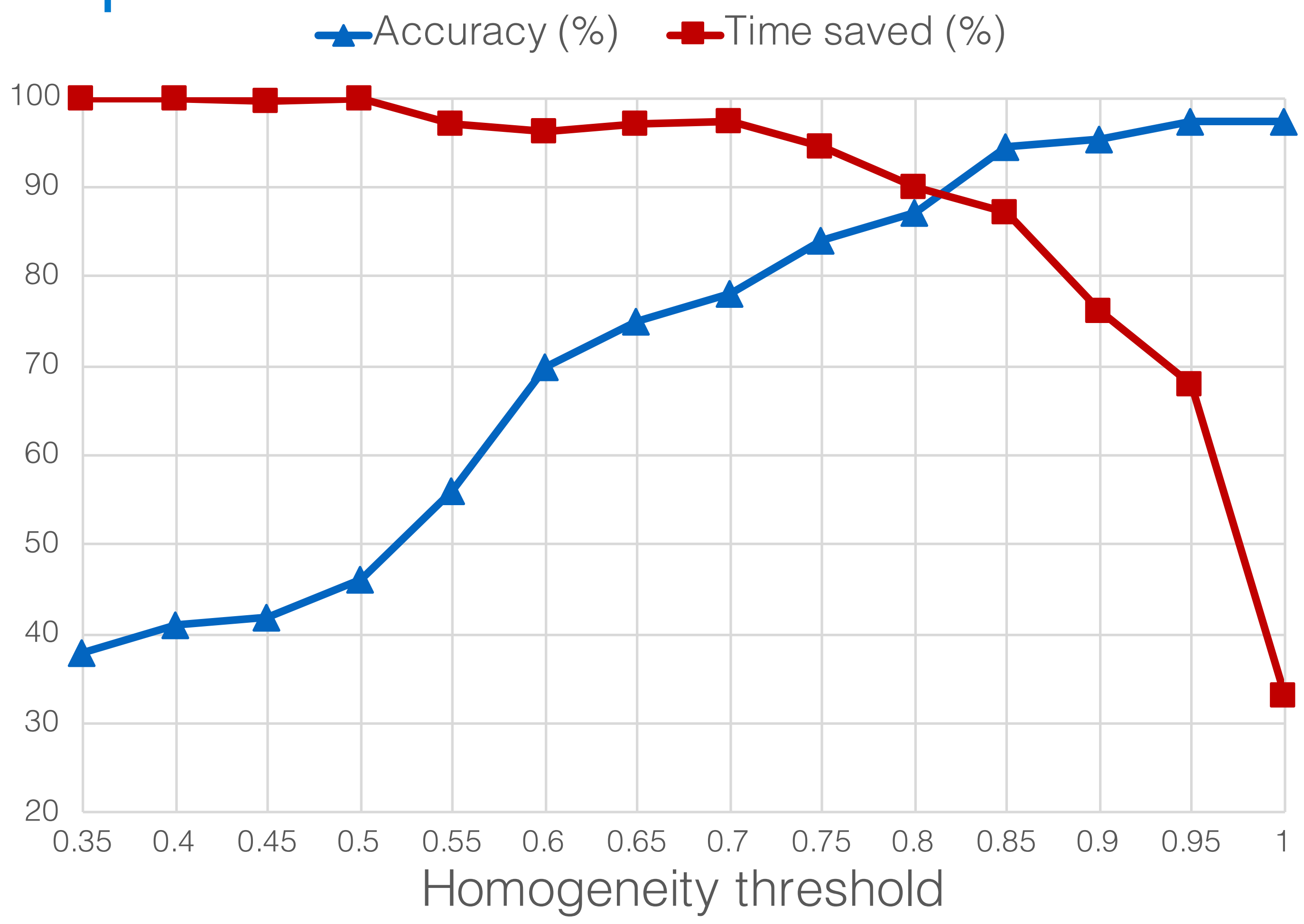
Audio workloads & datasets

- Speaker identification: TIMIT acoustic dataset

Reuse accuracy vs saving computation



Reuse accuracy vs saving computation



End-to-end performance

Application	Latency (ms)		Energy (mJ)		Accuracy loss (%)
	w/o	w/	w/o	w/	
Speaker Identification	13.1	4.2	30.4	9.8	3.2
Landmark Recognition	102.4	27.9	1315	110.7	5.0
Plant Recognition	269.6	99.8	3132	901.4	4.7

End-to-end performance

Application	Latency (ms)		Energy (mJ)		Accuracy loss (%)
	w/o	w/	w/o	w/	
Speaker Identification	13.1	**4.2**	30.4	9.8	3.2
Landmark Recognition	102.4	**27.9**			
Plant Recognition	269.6	**99.8**	3132	901.4	4.7

Over 3x latency reduction

End-to-end performance

Application	Latency (ms)		Energy (mJ)		Accuracy loss (%)
	w/o	w/	w/o	w/	
Speaker [illegible]	[illegible]	[illegible]	30.4	**9.8**	3.2
[illegible] Recognition	102.4	27.9	1315	**110.7**	5.0
Plant Recognition	269.6	99.8	3132	**901.4**	4.7

Up to 10x battery usage reduction

End-to-end performance

Application	Latency (ms)		Energy (mJ)		Accuracy loss (%)
	w/o	w/	w/o	w/	
Speaker Identificat					**3.2**
Landmark Recognition	102.4	27.9	1315	110.7	**5.0**
Plant Recognition	269.6	99.8	3132	901.4	**4.7**

Less than 5% accuracy loss

Conclusion

- FoggyCache: cross-device approximate computation reuse
    - Effectively eliminates fuzzy redundancy
- Approximate computation reuse
    - Promising new direction for optimizations
    - Algorithms are applicable to other scenarios

Thank you