#### Feature Indexing CSE 40537/60537 Biometrics





### Today you will...

#### Get to know Methods of feature indexing for **Biometric identification.**





## What is Biometrics?



#### 7 billion people

Who is this person? *(Identification)* Is this person Jane Doe? *(Verification)* 

Biometrics aims at *identifying* or *verifying* the claimed or denied identity of an individual based on their *physical*, *chemical* or *behavioral* traits.



## **Biometric Systems**

#### **Verification Modules**





#### **Biometric Systems** template database 2 template feature sensor 1. ID presentation





#### **Biometric Verification**

No need for complex feature indexing.





### **Biometric Verification**

No need for complex feature indexing.

Use unique person's ID as index (or hash function input).

Retrieval of features in constant time.

000
000
000
000





## **Biometric Systems**

#### **Identification Modules**









How to retrieve *k*-nearest features to compose gallery?

Need for more complex indexing.

**Retrieval of features** as quick as possible.

### **Biometric Identification**





How to retrieve k-nearest features to compose gallery?

Need for more complex indexing.

Retrieval of features as quick as possible.

#### **Biometric Identification**





#### **Level-1 Features**

#### **Usage of Singular Points and Core**



loop



delta



Jain, Ross, and Nadakum Introduction to Biometrics Springer Books, 2011



#### **Level-1 Features**

#### **Usage of Singular Points and Core**



plain arch





tented arch

left loop

Jain, Ross, and Nadakumar Introduction to Biometrics Springer Books, 2011

right loop



whorl

twin loop







#### **Level-1 Features**

**FBI** Automated Fingerprint Identification system (AFIS) More than 200 million dactyloscopy cards. Varied quality of samples.

Thanks to fingerprint classification through level-1 features, this time is reduced to 20 min.

right loop left loop plain arch whorl (2.7%)(27.9%)(31.7%)(33.8%)central pocket tented arch double loop accidental (2.9%)

Henry's features, an alternative classification of level-1 features with 8 classes.



#### Level-1 Features

FBI Automated Fingerprint Identification system (AFIS) More than 200 million dactyloscopy cards. Varied quality of samples.

And a computer-based solution can do it in seconds, benefitting from the same features.

#### 

Henry's features, an alternative classification of level-1 features with 8 classes.



#### **Iris Identification**



#### 2048 bits IrisCode





#### **Iris Identification**

#### **Face Identification**



#### 2048 bits IrisCode

#### 512D ArcFace embedding



#### **Iris Identification**

#### **Face Identification**



How to retrieve *k*-nearest features to compose gallery?

Need for more complex indexing.

Retrieval of features as quick as possible.





#### Brute Force Search





#### Brute Force Search

What is the computational complexity? Linear: *O(n)*, where *n* is the number of features.

How to reduce it?

queryqueryqueryqueryqueryqueryqueryqueryqueryquery





## Early Stop Search

a realure that is close enough.	query	
Stop when you find	query	
complexity?	query	
How to reduce	query	



Dr. Andrey Kuehlkamp mentioned this for iris identification.





How to reduce complexity?

Curves determined by index mapping functions that pass once through every point of an N-dimensional space.



(a)



(e)

#### 2D space examples





6

UNIVERSITY OF **IOTRE DAME** 



How to reduce complexity?

Curves determined by index mapping functions that pass once through every point of an *N*-dimensional space.



#### 2D space examples

#### Hilbert curves



How to reduce complexity?

Curves determined by index mapping functions that pass once through every point of an *N*-dimensional space.

The mapping functions are executed in constant time, w.r.t. the number of features.



#### 3D space examples

#### Hilbert curves





How to reduce complexity?

The curves are 1D and the elements indexed by them are "sorted" in an *approximation* of their distances in the original space.

If the curve is used as a binary tree, an approximation of the k-nearest elements can be obtained in *O(log(n))*, where *n* is the number of features.





How to reduce complexity?

Cluster the features and limit the k-nearest search to one or a couple of clusters.

There will be less elements to consider

Source: https://people.csail.mit.edu/ dsontag/courses/ml12/slides/lecture14.pdf



1.0

How to reduce complexity?

Cluster the features and limit the k-nearest search to one or a couple of clusters.

There will be less elements to consider

Source: https://people.csail.mit.edu/ dsontag/courses/ml12/slides/lecture14.pdf



.



K-Means

#### Select K random features as cluster centers.







K-Means

#### Assign features to closes cluster centers.







K-Means

#### Update the cluster centers by taking the **means** of each cluster.







K-Means

#### Repeat until convergence.







**K-Means** 

What are the limitations of this approach?

What is the ideal number of clusters?

Complexity of building clusters: O(Kn) in each step until convergence.

Clustering is *offline*: i.e., it does not happen at feature querying time.





#### Variation: K-medoids

Instead of using *means* as the cluster centers, use *medians*, which are actual existing features.

![](_page_33_Figure_3.jpeg)

![](_page_33_Picture_4.jpeg)

K-dimensional trees: For K times Split one feature dimension into two halves.

![](_page_34_Picture_3.jpeg)

#### 2D-features toy case

![](_page_34_Figure_5.jpeg)

![](_page_34_Picture_8.jpeg)

K-dimensional trees: For K times Split one feature dimension into two partitions using medians.

![](_page_35_Picture_3.jpeg)

![](_page_35_Picture_4.jpeg)

#### 2D-features toy case

![](_page_35_Figure_6.jpeg)

![](_page_35_Picture_9.jpeg)

K-dimensional trees: For K times Split one feature dimension into two partitions using medians.

![](_page_36_Picture_3.jpeg)

![](_page_36_Picture_4.jpeg)

![](_page_36_Figure_5.jpeg)

![](_page_36_Figure_6.jpeg)

![](_page_36_Picture_7.jpeg)

K-dimensional trees: For K times Split one feature dimension into two partitions using medians.

![](_page_37_Picture_3.jpeg)

![](_page_37_Picture_4.jpeg)

#### 2D-features toy case

![](_page_37_Figure_6.jpeg)

![](_page_37_Picture_7.jpeg)

K-dimensional trees: For K times Split one feature dimension into two partitions using medians.

![](_page_38_Picture_3.jpeg)

![](_page_38_Picture_4.jpeg)

![](_page_38_Figure_5.jpeg)

![](_page_38_Picture_7.jpeg)

K-dimensional trees: For K times Split one feature dimension into two partitions using medians.

Complexity to build: O(n log(n))

Building is *offline*: i.e., it does not happen at feature querying time.

![](_page_39_Picture_5.jpeg)

![](_page_39_Figure_6.jpeg)

![](_page_39_Picture_8.jpeg)

How to obtain 3-nearest neighbors?

![](_page_40_Picture_3.jpeg)

![](_page_40_Picture_4.jpeg)

#### 2D-features toy case

![](_page_40_Figure_6.jpeg)

![](_page_40_Picture_7.jpeg)

![](_page_41_Picture_3.jpeg)

![](_page_41_Picture_4.jpeg)

![](_page_41_Figure_5.jpeg)

![](_page_41_Picture_8.jpeg)

![](_page_42_Picture_3.jpeg)

![](_page_42_Picture_4.jpeg)

![](_page_42_Figure_5.jpeg)

![](_page_42_Picture_7.jpeg)

![](_page_43_Picture_3.jpeg)

![](_page_43_Picture_4.jpeg)

![](_page_43_Figure_5.jpeg)

![](_page_43_Picture_7.jpeg)

How to obtain 3-nearest neighbors?

![](_page_44_Picture_3.jpeg)

![](_page_44_Picture_4.jpeg)

![](_page_44_Figure_5.jpeg)

![](_page_44_Picture_7.jpeg)

![](_page_45_Picture_3.jpeg)

![](_page_45_Picture_4.jpeg)

![](_page_45_Figure_5.jpeg)

![](_page_45_Picture_6.jpeg)

![](_page_45_Picture_7.jpeg)

How to obtain 3-nearest neighbors?

![](_page_46_Picture_3.jpeg)

No changes in 3-nearest, so stop.

![](_page_46_Picture_5.jpeg)

#### 2D-features toy case

![](_page_46_Figure_7.jpeg)

![](_page_46_Picture_10.jpeg)

![](_page_47_Figure_2.jpeg)

![](_page_47_Figure_3.jpeg)

Toy Case (6D features, reality: 512D for faces)

![](_page_47_Picture_5.jpeg)

*M* features

![](_page_47_Picture_7.jpeg)

![](_page_47_Picture_8.jpeg)

Toy Case (6D features, reality: 512D for faces)

How to reduce	ID 1	
siza?	ID 1	
	ID 2	
State-of-the-art feature indexing.	ID 2	
	ID 2	
	ID 3	
	ID 3	
1. Start with a	ID 4	
coarse quantizer.		

![](_page_48_Figure_3.jpeg)

![](_page_48_Picture_4.jpeg)

. . .

49

coarse quantizer

![](_page_48_Picture_6.jpeg)

![](_page_48_Picture_7.jpeg)

![](_page_48_Picture_8.jpeg)

![](_page_48_Picture_9.jpeg)

Toy Case (6D features, reality: 512D for faces)

How to reduce	ID 1	
siza?	ID 1	
	ID 2	
State-of-the-art feature indexing.	ID 2	
	ID 2	
	ID 3	
	ID 3	
1. Start with a	ID 4	
coarse quantizer.		

![](_page_49_Figure_3.jpeg)

![](_page_49_Picture_4.jpeg)

![](_page_49_Picture_5.jpeg)

![](_page_49_Picture_6.jpeg)

![](_page_49_Picture_7.jpeg)

![](_page_49_Picture_8.jpeg)

How to reduce	ID 1	
siza?	ID 1	
	ID 2	
State-of-the-art feature indexing.	ID 2	
	ID 2	
	ID 3	
	ID 3	
1. Start with a	ID 4	
coarse quantizer.		

![](_page_50_Figure_3.jpeg)

![](_page_50_Picture_4.jpeg)

How to reduce	ID 1	
siza?	ID 1	
	ID 2	
State-of-the-art feature indexing.	ID 2	
	ID 2	
	ID 3	
	ID 3	
1. Start with a	ID 4	
coarse quantizer.		

![](_page_51_Figure_3.jpeg)

![](_page_51_Picture_4.jpeg)

How to reduce	ID 1	
sizo?	ID 1	
	ID 2	
State_of_the_art feature	ID 2	
indovina	ID 2	
muexing.	ID 3	
2 Compute <b>residuals</b>	ID 3	
(difference) between	ID 4	
(unierences) between		
teatures and their		
respective coarse	ID P	
centroids.		М

![](_page_52_Figure_3.jpeg)

![](_page_52_Picture_4.jpeg)

Toy Case

How to reduce	ID 1	
sizo?	ID 1	
	ID 2	
State_of_the_art feature	ID 2	
indovina	ID 2	
muexing.	ID 3	
2 Compute <b>residuals</b>	ID 3	
(difference) between	ID 4	
(unierences) between		
teatures and their		
respective coarse	ID P	
centroids.		М

![](_page_53_Figure_3.jpeg)

Toy Case

How to reduce	ID 1
nizo?	ID 1
3120:	ID 2
State_of_the_art feature	ID 2
indexing.	ID 2
	ID 3
	ID 3
3. Reduce the	ID 4
dimensionality of	
residuals with	
<b>Product Quantization</b>	ID P
	Λ <i>Λ</i>

![](_page_54_Figure_3.jpeg)

![](_page_54_Picture_4.jpeg)

M residuals

![](_page_54_Picture_6.jpeg)

Toy Case

How to reduce	ID 1
nizo?	ID 1
3120:	ID 2
State_of_the_art feature	ID 2
indexing.	ID 2
	ID 3
	ID 3
3. Reduce the	ID 4
dimensionality of	
residuals with	
<b>Product Quantization</b>	ID P
	Λ <i>Λ</i>

![](_page_55_Figure_3.jpeg)

**Product Quantization** 

![](_page_55_Figure_5.jpeg)

*M* residuals D dimensions

![](_page_55_Picture_7.jpeg)

*M* residuals

![](_page_55_Picture_9.jpeg)

![](_page_55_Figure_11.jpeg)

Toy Case

How to reduce	ID 1
size?	ID 1
3120:	ID 2
State_of_the_art feature	ID 2
indovina	ID 2
muexing.	ID 3
	ID 3
3. Reduce the	ID 4
dimensionality of	
residuals with	
<b>Product Quantization.</b>	ID P
	M

![](_page_56_Figure_3.jpeg)

**Product Quantization** 

![](_page_56_Figure_5.jpeg)

![](_page_56_Picture_6.jpeg)

![](_page_56_Picture_7.jpeg)

![](_page_56_Picture_9.jpeg)

Toy Case

How to reduce	ID 1
size?	ID 1
3120:	ID 2
Stata_of_tho_art foaturo	ID 2
indovina	ID 2
muexing.	ID 3
	ID 3
3. Reduce the	ID 4
dimensionality of	
residuals with	
<b>Product Quantization.</b>	ID P
	M

![](_page_57_Figure_3.jpeg)

**Product Quantization** 

![](_page_57_Figure_5.jpeg)

![](_page_57_Picture_6.jpeg)

residuals

![](_page_57_Picture_8.jpeg)

![](_page_57_Figure_10.jpeg)

Toy Case

How to reduce	ID 1
sizo?	ID 1
	ID 2
State of the art feature	ID 2
indoving	ID 2
muexing.	ID 3
	ID 3
3. Reduce the	ID 4
dimensionality of	
residuals with	
<b>Product Quantization.</b>	
	M

![](_page_58_Figure_3.jpeg)

Product Quantization

![](_page_58_Figure_5.jpeg)

![](_page_58_Picture_6.jpeg)

![](_page_58_Picture_7.jpeg)

Toy Case

How to reduce	ID 1
size?	ID 1
3120:	ID 2
Stata_of_tho_art foaturo	ID 2
indovina	ID 2
muexing.	ID 3
	ID 3
3. Reduce the	ID 4
dimensionality of	
residuals with	
<b>Product Quantization.</b>	ID P
	M

![](_page_59_Figure_3.jpeg)

**Product Quantization** 

![](_page_59_Figure_5.jpeg)

residuals

. . .

60

How to reduce	ID 1	
siza?	ID 1	
5120:	ID 2	
Stata_of_tha_art faatura	ID 2	
indoving	ID 2	
muexing.	ID 3	
	ID 3	
1 Appand the product	ID 4	
quantized residuals to		
an inverted file index.	ID P	

![](_page_60_Figure_4.jpeg)

![](_page_60_Picture_5.jpeg)

How to reduce	ID 1	
sizo?	ID 1 📒	
3120:	ID 2	
State_of_the_art feature	ID 2	
indoving	ID 2	
muexing.	ID 3	
	ID 3	
1 Annond the product	ID 4	
quantized residuals to		
an inverted file index.	ID P	

![](_page_61_Figure_4.jpeg)

![](_page_61_Picture_5.jpeg)

How to reduce	ID 1	
sizo?	ID 1	
	ID 2	
Stata_of_tha_art faatura	ID 2	
indoving	ID 2	
maexing.	ID 3	
	ID 3	
1 Annond the product	ID 4	
quantized residuals to		
an inverted file index.	ID P	

![](_page_62_Figure_4.jpeg)

![](_page_62_Picture_5.jpeg)

How to reduce	ID 1	
siza?	ID 1	
	ID 2	
State_of_the_art feature	ID 2	
indoving	ID 2	
maexing.	ID 3	
	ID 3	
1 Appand the product	ID 4	
quantized residuals to		
an inverted file index.	ID P	

![](_page_63_Figure_4.jpeg)

Toy Case (6D features, reality: 512D for faces)

![](_page_64_Figure_2.jpeg)

an inverted file index.

Toy Case (6D features, reality: 512D for faces)

![](_page_65_Figure_2.jpeg)

an inverted file index.

How to reduce size?

State-of-the-art feature indexing.

Usage example: Indexing.

![](_page_66_Figure_4.jpeg)

![](_page_66_Picture_5.jpeg)

![](_page_66_Figure_6.jpeg)

How to reduce size?

State-of-the-art feature indexing.

Usage example: **Retrieving k-nearest.** 

![](_page_67_Figure_4.jpeg)

Source: Jegou et al. Product quantization for nearest neighbor search IEEE T-PAMI 2010

![](_page_67_Picture_6.jpeg)

![](_page_67_Figure_7.jpeg)

How to reduce size?

State-of-the-art feature indexing.

#### Available implementation.

#### *⋴* Faiss

Faiss is a library for efficient similarity search and clustering of dense vectors. It contains algorithms that search in sets of vectors of any size, up to ones that possibly do not fit in RAM. It also contains supporting code for evaluation and parameter tuning. Faiss is written in C++ with complete wrappers for Python/numpy. Some of the most useful algorithms are implemented on the GPU. It is developed primarily at Facebook Al Research.

![](_page_68_Figure_6.jpeg)

Product $\vee$ Team Enter	erprise Explore $\smallsetminus$ Marketplace Pricing $\smallsetminus$	Search	Sign in Sign up
facebookresearch / fa	Public 가 Pull requests 8 및 Discussions	<ul> <li>Notifications</li> <li>Actions</li></ul>	♀     Fork     2.6k     ☆     Star     16.6k       ojects     4     □     Wiki
° main - ິະ 13 branche	es ເ∿ 17 tags Go to x-github-bot Auto ··· × 1806c6a yesterday	o file Code →	About A library for efficient similarity search and clustering of dense vectors.
<ul> <li>.circleci</li> <li>.github</li> <li>benchs</li> <li>c_api</li> <li>cmake</li> <li>conda</li> </ul>	Add IndexNSGPQ and IndexNSGSQ (#2218) Change default branch references from master contrib clustering module (#2217) Generalize DistanceComputer for flat indexes ( Move from TravisCI to CircleCI (#1315) Fix packaging (#2121)	last month 7 months ago last month 11 days ago 2 years ago 4 months ago	<ul> <li>✓ faiss.ai</li> <li>✓ Readme</li> <li>✓ MIT License</li> <li>✓ Code of conduct</li> <li>✓ 16.6k stars</li> <li>✓ 443 watching</li> <li>✓ 2.6k forks</li> </ul>
contrib demos faiss misc tests	<ul> <li>contrib clustering module (#2217)</li> <li>Add NNDescent to faiss (#1654)</li> <li>Automatic type conversions for Python API (#2</li> <li>Enable clang-format + autofix.</li> <li>Automatic type conversions for Python API (#2</li> </ul>	last month 13 months ago yesterday 13 months ago yesterday	Releases 13 S Faiss 1.7.2 Latest on Jan 10 + 12 releases

#### https://github.com/facebookresearch/faiss

![](_page_68_Picture_9.jpeg)

#### Content

![](_page_69_Figure_2.jpeg)

**Basics** Concepts **Metrics** Metric implementation

![](_page_69_Picture_4.jpeg)

![](_page_69_Picture_5.jpeg)

![](_page_69_Picture_6.jpeg)

**Core Traits** (3) Concepts Evaluation Assignments

### S'up Next?

![](_page_69_Picture_9.jpeg)

**Alternative Traits and Fusion** Concepts

**Invited Talks** (2) State of the art Future work

![](_page_69_Picture_13.jpeg)

- **Baseline implementation**

![](_page_69_Picture_19.jpeg)

#### Content

![](_page_70_Figure_2.jpeg)

Basics Concepts Metrics Metric implementation

![](_page_70_Picture_4.jpeg)

![](_page_70_Picture_5.jpeg)

![](_page_70_Picture_6.jpeg)

**Core Traits** (3) Concepts **Baseline implementation** Evaluation Assignments

### S'up Next?

![](_page_70_Picture_11.jpeg)

**Alternative Traits and Fusion** Concepts

![](_page_70_Picture_13.jpeg)

**Invited Talks** (2) State of the art Future work

![](_page_70_Picture_15.jpeg)

![](_page_70_Picture_17.jpeg)