

# Finding Similar Items I

Alexander Schönhuth



Bielefeld University  
April 22, 2021

# TODAY

## *Announcements*

- ▶ Lecture will be *recorded*, edited and posted (as usual)
- ▶ From today, topics are relevant for exam
- ▶ *Reminder*: Please assign yourself to a group in the LernraumPlus
- ▶ Groups were supposed to be (approximately) 5 people, to collectively submit solutions and present in tutorials

## *Learning Goals*

- ▶ Turning documents into sets  $\rightarrow$  shingles
- ▶ Computing the similarity of sets  $\rightarrow$  minhashing

## *Finding Similar Items: Introduction*

# FINDING SIMILAR ITEMS

Fundamental problem in data mining: retrieve pairs of similar elements of a dataset.

## Applications

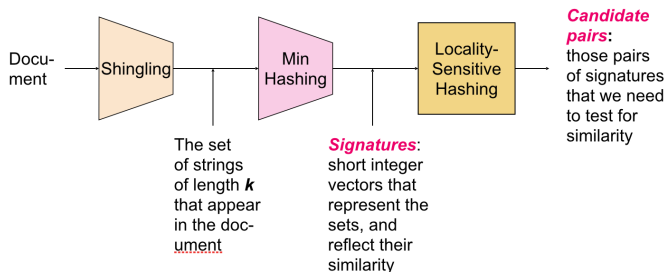
- ▶ Detecting plagiarism in a set of documents
- ▶ Identifying near-identical mirror pages during web searches
- ▶ Identifying documents from the same author
- ▶ *Collaborative Filtering*
  - ▶ Online Purchases (Amazon: suggestions based on 'similar' customers)
  - ▶ Movie Ratings (Netflix: suggestions based on 'similar' users)

# ISSUES

Consider a dataset of  $N$  items, for example:  $N$  webpages or  $N$  text documents.

- ▶ Comparing all items requires  $O(N^2)$  runtime.
  - ▶ Ok for small  $N$ .
  - ▶ If  $N \approx 10^6$ , we have  $10^{12}$  comparisons. Maybe not OK!
- ▶ How to efficiently compute similarity if items themselves are large?
- ▶ Similarity works well for sets of items. How to turn data into sets of items?

# OVERVIEW



From [mmds.org](http://mmds.org)

- ▶ *Shingling*: turning text files into sets
- ▶ *Minhashing*: computing similarity for large sets
- ▶ *Locality Sensitive Hashing*: avoids  $O(N^2)$  comparisons by determining candidate pairs

*Shingles*  
—  
*Turning Documents into Sets*

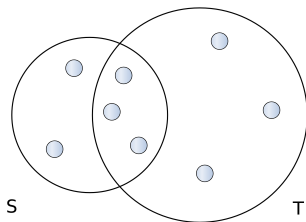
# JACCARD SIMILARITY

## DEFINITION [JACCARD SIMILARITY]

Consider two sets  $S$  and  $T$ . The *Jaccard similarity*  $\text{SIM}(S, T)$  is defined as

$$\text{SIM}(S, T) = \frac{|S \cap T|}{|S \cup T|} \quad (1)$$

the ratio of elements in the intersection and in the union of  $S$  and  $T$ .



$$\text{SIM}(S, T) = \frac{3}{8}$$



# SHINGLES: DEFINITION

- ▶ Document = large string of characters
- ▶ *k*-shingle: a substring of a particular length *k*
- ▶ *Idea*: A document is set of *k*-shingles
- ▶ *Example*: document = "acadacc", *k*-shingles:

$$\{ac, ad, ca, cc, da\}$$

- ▶ We can now compute *Jaccard similarity* for two documents by considering them as sets of shingles.
- ▶ *Example*: documents  $D_1 = \text{"abcd"}$ ,  $D_2 = \text{"dbcd"}$  using 2-shingles yields  $D_1 = \{ab, bc, cd\}$ ,  $D_2 = \{bc, cd, db\}$ , so 
$$\text{SIM}(D_1, D_2) = \frac{|\{bc, cd\}|}{|\{ab, bc, cd, db\}|} = 2/4 = 1/2$$

# SHINGLES: DEFINITION

- ▶ Issue: Determining right size of  $k$ .
  - ▶  $k$  large enough such that any particular  $k$ -shingle appears in document with low probability ( $k = 5$ , yielding  $256^5$  different shingles on 256 different characters, ok for emails)
  - ▶ too large  $k$  yields too large universe of elements (example:  $k = 9$  means  $256^9 = (2^8)^9 = 2^{72}$  on the order of number of atoms in the universe)
- ▶ Solution if necessary  $k$  is too large: hash shingles to buckets, such that buckets are evenly covered, and collisions are rare
- ▶ We would like to compute Jaccard similarity for pairs of sets
- ▶ *But*: even when hashed, size of the universe of elements (= # buckets when hashed) may be prohibitive to do that fast
- ▶ What to do?

*Minhashing*  
—  
*Rapidly Computing Similarity of Sets*

# SETS AS BITVECTORS

- ▶ Representing sets as bitvectors
  - ▶ Length of bitvectors is size of universal set
  - ▶ For example, when hashed, length of bitvector = number of buckets
  - ▶ Entries zero if *element not in set*, one if *element in set*
- ▶ Does not reflect to really store the sets, but nice visualization

# SETS AS BITVECTORS: THE CHARACTERISTIC MATRIX

DEFINITION [CHARACTERISTIC MATRIX]

Given  $C$  sets over a universe  $R$ , the *characteristic matrix*  $M \in \{0, 1\}^{|R| \times |C|}$  is defined to have entries

$$M(r, c) = \begin{cases} 0 & \text{if } r \notin c \\ 1 & \text{if } r \in c \end{cases} \quad (2)$$

for  $r \in R, c \in C$ .

<i>Element</i>	$S_1$	$S_2$	$S_3$	$S_4$
<i>a</i>	1	0	0	1
<i>b</i>	0	0	1	0
<i>c</i>	0	1	0	1
<i>d</i>	1	0	1	1
<i>e</i>	0	0	1	0

*Characteristic matrix* of four sets ( $S_1, S_2, S_3, S_4$ ) over universal set  $\{a, b, c, d, e\}$

From [mmds.org](http://mmds.org)

# PERMUTATIONS

## DEFINITION [BIJECTION, PERMUTATION]

- ▶ A *bijection* is a map  $\pi : S \rightarrow S$  such that
  - ▶  $\pi(x) = \pi(y)$  implies  $x = y$  ( $\pi$  is *injective*)
  - ▶ For all  $y \in S$  there is  $x \in S$  such that  $\pi(x) = y$  ( $\pi$  is *surjective*)
- ▶ A *permutation* is a bijection

$$\pi : \{1, \dots, m\} \rightarrow \{1, \dots, m\} \quad (3)$$

*Example:* A permutation on  $\{1, 2, 3, 4, 5\}$  may map

$$1 \rightarrow 4, 2 \rightarrow 3, 3 \rightarrow 1, 4 \rightarrow 5 \text{ and } 5 \rightarrow 2$$

# PERMUTING ROWS OF CHARACTERISTIC MATRIX

<i>Element</i>	$S_1$	$S_2$	$S_3$	$S_4$
<i>a</i>	1	0	0	1
<i>b</i>	0	0	1	0
<i>c</i>	0	1	0	1
<i>d</i>	1	0	1	1
<i>e</i>	0	0	1	0

<i>Element</i>	$S_1$	$S_2$	$S_3$	$S_4$
<i>b</i>	0	0	1	0
<i>e</i>	0	0	1	0
<i>a</i>	1	0	0	1
<i>d</i>	1	0	1	1
<i>c</i>	0	1	0	1

A characteristic matrix of four sets ( $S_1, S_2, S_3, S_4$ ) over universal set  $\{a, b, c, d, e\}$  and a permutation of its rows  $1 \rightarrow 3, 2 \rightarrow 1, 3 \rightarrow 5, 4 \rightarrow 4, 5 \rightarrow 2$

# MINHASH - DEFINITION

Given

- ▶ a characteristic matrix with  $m$  rows and a column  $S$
- ▶ a permutation  $\pi$  on the rows, that is  $\pi : \{1, \dots, m\} \rightarrow \{1, \dots, m\}$  is a bijection

DEFINITION [MINHASH]

The *minhash* function  $h_\pi$  on  $S$  is defined by

$$h_\pi(S) = \min_{i \in \{1, \dots, m\}} \{\pi(i) \mid S[i] = 1\}$$



# MINHASH - DEFINITION

## DEFINITION [MINHASH]

The *minhash* function  $h_\pi$  on  $S$  is defined by

$$h_\pi(S) = \min_{i \in \{1, \dots, m\}} \{\pi(i) \mid S[i] = 1\}$$

## EXPLANATION

The minhash of a column  $S$  relative to permutation  $\pi$  is

- ▶ after reordering rows according to the permutation  $\pi$
- ▶ the first row in which a one in  $S$  appears

# MINHASH - EXAMPLE

EXAMPLE

Let

- ▶ 1 corresponds to  $a$ , 2 to  $b$ , ...
- ▶  $\pi : 1 \rightarrow 3, 2 \rightarrow 1, 3 \rightarrow 5, 4 \rightarrow 4, 5 \rightarrow 2$  and

<i>Element</i>	$S_1$	$S_2$	$S_3$	$S_4$
$b$	0	0	1	0
$e$	0	0	1	0
$a$	1	0	0	1
$d$	1	0	1	1
$c$	0	1	0	1

$$h_\pi(S_1) = 3, h_\pi(S_2) = 5, h_\pi(S_3) = 1, h_\pi(S_4) = 3$$

# MINHASHING AND JACCARD SIMILARITY

Given

- ▶ two columns (sets)  $S_1, S_2$  of a characteristic matrix
- ▶ a randomly picked permutation  $\pi$  on the rows (on  $\{1, \dots, m\}$ )

THEOREM [MINHASH AND JACCARD SIMILARITY]:

The probability that  $h_\pi(S_1) = h_\pi(S_2)$  is  $\text{SIM}(S_1, S_2)$ .

# MINHASH AND JACCARD SIMILARITY - PROOF

THEOREM [MINHASH AND JACCARD SIMILARITY]:  
The probability that  $h_\pi(S_1) = h_\pi(S_2)$  is  $\text{SIM}(S_1, S_2)$ .

PROOF.

Distinguish three different classes of rows:

- ▶ *Type X rows* have a 1 in both  $S_1, S_2$
- ▶ *Type Y rows* have a 1 in only one of  $S_1, S_2$
- ▶ *Type Z rows* have a 0 in both  $S_1, S_2$

Let  $x$  be the number of type X rows and  $y$  the number of type Y rows.

- ▶ So  $x = |S_1 \cap S_2|$  and  $x + y = |S_1 \cup S_2|$
- ▶ Hence

$$\text{SIM}(S_1, S_2) = \frac{|S_1 \cap S_2|}{|S_1 \cup S_2|} = \frac{x}{x + y} \quad (4)$$

# MINHASH AND JACCARD SIMILARITY - PROOF

PROOF. (CONT.)

- ▶ Consider the *probability* that  $h(S_1) = h(S_2)$
- ▶ Imagine rows to be permuted randomly, and proceed from the top
- ▶ The probability to encounter type X before type Y is

$$\frac{x}{x + y} \tag{5}$$

- ▶ If first non type Z row is type X, then  $h(S_1) = h(S_2)$
- ▶ If first non type Z row is type Y, then  $h(S_1) \neq h(S_2)$
- ▶ So  $h(S_1) = h(S_2)$  happens with probability (5), which by (4) concludes the proof.



# MINHASH - INTERMEDIATE SUMMARY / EXPANSION OF IDEA

- ▶ Computing a minhash means turning a set into one number
- ▶ For different sets, numbers agree with probability equal to their Jaccard similarity.
- ▶ Can we expand on this idea? Can we compute (ensembles of) numbers that enable us to determine their Jaccard similarity?
- ▶ Immediate idea: compute several minhashes. The fraction of times the minhashes of two sets agree equals their Jaccard similarity.
- ▶ Several sufficiently well chosen minhashes yield a *minhash signature*.

# MINHASH SIGNATURES

Consider

- ▶ the  $m$  rows of the characteristic matrix
- ▶  $n$  permutations  $\{1, \dots, m\} \rightarrow \{1, \dots, m\}$
- ▶ the corresponding *minhash* functions  $h_1, \dots, h_n : \{0, 1\}^m \rightarrow \{1, \dots, m\}$
- ▶ and a particular column  $S \in \{0, 1\}^m$   
    ☞  $h_i(S) \in \{1, \dots, m\}$  for any  $1 \leq i \leq n$

DEFINITION [MINHASH SIGNATURE]

The *minhash signature*  $SIG_S$  of  $S$  given  $h_1, \dots, h_n$  is the array

$$[h_1(S), \dots, h_n(S)] \in \{1, \dots, m\}^n$$

# MINHASH SIGNATURES

## DEFINITION [MINHASH SIGNATURE]

The *minhash signature*  $SIG_S$  of  $S$  given  $h_1, \dots, h_n$  is the array

$$[h_1(S), \dots, h_n(S)] \in \{1, \dots, m\}^n$$

*Meaning:* Computing the minhash signature for a column  $S$  turns

- ▶ the binary-valued array of length  $m$  that represents  $S$   
 $\leftrightarrow S \in \{0, 1\}^m$
- ▶ into an  $m$ -valued array of length  $n$   
 $\leftrightarrow [h_1(S), \dots, h_n(S)] \in \{1, \dots, m\}^n$

Because  $n < m$  (often  $n \ll m$ ), the minhash signature is a *reduced representation of a set*.



# SIGNATURE MATRIX

Consider a characteristic matrix, and  $n$  permutations  $h_1, \dots, h_n$ .

DEFINITION [SIGNATURE MATRIX]

The signature matrix  $SIG$  is a matrix with  $n$  rows and as many columns as the characteristic matrix (i.e. the number of sets), where entries  $SIG_{ij}$  are defined by

$$SIG_{ij} = h_i(S_j) \tag{6}$$

where  $S_j$  refers to the  $j$ -th column in the characteristic matrix.

# SIGNATURE MATRICES: FACTS

Let  $M$  be a signature matrix.

- ▶ Because usually  $n \ll m$ , that is  $n$  is much smaller than  $m$ , a signature matrix is much smaller than the original characteristic matrix.
- ▶ The probability that  $SIG_{ij_1} = SIG_{ij_2}$  for two sets  $S_{j_1}, S_{j_2}$  equals the Jaccard similarity  $SIM(S_{j_1}, S_{j_2})$
- ▶ The expected number of rows where columns  $j_1, j_2$  agree, divided by  $n$ , is  $SIM(S_{j_1}, S_{j_2})$ .

# SIGNATURE MATRICES: ISSUES

## *Issue:*

- ▶ For large  $m$ , it is time-consuming / storage-intense to determine permutations

$$\pi : \{1, \dots, m\} \rightarrow \{1, \dots, m\}$$

- ▶ Re-sorting rows relative to a permutation is even more expensive

## *Solution:*

- ▶ Instead of permutations, use hash functions (watch the index shift!)

$$h : \{0, \dots, m - 1\} \rightarrow \{0, \dots, m - 1\}$$

- ▶ Likely, a hash function is not a bijection, so at times
  - ▶ places two rows in the same bucket
  - ▶ leaves other buckets empty
- ▶ Effects are negligible for our purposes, however

# COMPUTING SIGNATURE MATRICES IN PRACTICE

- ▶ Consider  $n$  hash functions

$$h_i : \{0, \dots, m-1\} \rightarrow \\ \{0, \dots, m-1\}, i = 1, \dots, n$$

- ▶ Let  $r$  and  $c$  index rows and columns in the characteristic matrix  $M \in \{0, 1\}^{m \times |C|}$
- ▶ So  $c$  also index columns, while  $i$  indexes rows in the signature matrix  $SIG \in \{1, \dots, m\}^{n \times |C|}$

```
for each  $c$  do
  for  $0 \leq i \leq n$  do
     $SIG(i, c) = \infty$ 
  end for
end for
for each row  $r$  do
  for each column  $c$  do
    if  $M(r, c) = 1$  then
      for  $i=1$  to  $n$  do
         $SIG(i, c) =$ 
           $\min(SIG(i, c), h_i(r))$ 
      end for
    end if
  end for
end for
```

## COMPUTING SIGNATURE MATRICES: EXAMPLE

<i>Row</i>	$S_1$	$S_2$	$S_3$	$S_4$	$x + 1 \pmod{5}$	$3x + 1 \pmod{5}$
0	1	0	0	1	1	1
1	0	0	1	0	2	4
2	0	1	0	1	3	2
3	1	0	1	1	4	0
4	0	0	1	0	0	3

Hash functions computed for a characteristic matrix, with rows indexed from 0 to 4

# COMPUTING SIGNATURE MATRICES IN PRACTICE

- ▶ Consider  $n$  hash functions  
 $h_i : \{0, \dots, m - 1\} \rightarrow$   
 $\{0, \dots, m - 1\}, i = 1, \dots, n$
- ▶ Let  $r$  and  $c$  index rows and columns in the characteristic matrix  $M \in \{0, 1\}^{m \times |C|}$
- ▶ So  $c$  also index columns, while  $i$  indexes rows in the signature matrix  $SIG \in \{1, \dots, m\}^{n \times |C|}$

```
for each  $c$  do
  for  $0 \leq i \leq n$  do
     $SIG(i, c) = \infty$ 
  end for
end for
for each row  $r$  do
  for each column  $c$  do
    if  $M(r, c) = 1$  then
      for  $i=1$  to  $n$  do
         $SIG(i, c) =$ 
           $\min(SIG(i, c), h_i(r))$ 
      end for
    end if
  end for
end for
```

# COMPUTING SIGNATURE MATRICES: EXAMPLE

<i>Row</i>	$S_1$	$S_2$	$S_3$	$S_4$	$x + 1 \pmod{5}$	$3x + 1 \pmod{5}$
0	1	0	0	1	1	1
1	0	0	1	0	2	4
2	0	1	0	1	3	2
3	1	0	1	1	4	0
4	0	0	1	0	0	3

Hash functions computed for a characteristic matrix, with rows indexed from 0 to 4

	$S_1$	$S_2$	$S_3$	$S_4$
$h_1$	$\infty$	$\infty$	$\infty$	$\infty$
$h_2$	$\infty$	$\infty$	$\infty$	$\infty$

Signature matrix  $SIG$ : after initialization

# COMPUTING SIGNATURE MATRICES IN PRACTICE

- ▶ Consider  $n$  hash functions  
 $h_i : \{0, \dots, m - 1\} \rightarrow$   
 $\{0, \dots, m - 1\}, i = 1, \dots, n$
- ▶ Let  $r$  and  $c$  index rows and columns in the characteristic matrix  $M \in \{0, 1\}^{m \times |C|}$
- ▶ So  $c$  also index columns, while  $i$  indexes rows in the signature matrix  $SIG \in \{1, \dots, m\}^{n \times |C|}$

```
for each  $c$  do
  for  $0 \leq i \leq n$  do
     $SIG(i, c) = \infty$ 
  end for
end for
for each row  $r$  do
  for each column  $c$  do
    if  $M(r, c) = 1$  then
      for  $i=1$  to  $n$  do
         $SIG(i, c) =$ 
           $\min(SIG(i, c), h_i(r))$ 
      end for
    end if
  end for
end for
```



# COMPUTING SIGNATURE MATRICES IN PRACTICE

- ▶ Consider  $n$  hash functions  
 $h_i : \{0, \dots, m - 1\} \rightarrow$   
 $\{0, \dots, m - 1\}, i = 1, \dots, n$
- ▶ Let  $r$  and  $c$  index rows and columns in the characteristic matrix  $M \in \{0, 1\}^{m \times |C|}$
- ▶ So  $c$  also index columns, while  $i$  indexes rows in the signature matrix  $SIG \in \{1, \dots, m\}^{n \times |C|}$

```
for each  $c$  do
  for  $0 \leq i \leq n$  do
     $SIG(i, c) = \infty$ 
  end for
end for
for each row  $r$  do
  // Iteration 1: first row
  for each column  $c$  do
    if  $M(r, c) = 1$  then
      for  $i=1$  to  $n$  do
         $SIG(i, c) =$ 
           $\min(SIG(i, c), h_i(r))$ 
      end for
    end if
  end for
  // End first row
end for
```

## COMPUTING SIGNATURE MATRICES: EXAMPLE

Row	$S_1$	$S_2$	$S_3$	$S_4$	$x + 1 \pmod{5}$	$3x + 1 \pmod{5}$
0	1	0	0	1	1	1
1	0	0	1	0	2	4
2	0	1	0	1	3	2
3	1	0	1	1	4	0
4	0	0	1	0	0	3

Hash functions computed for a characteristic matrix, with rows indexed from 0 to 4

**First iteration:** row # 0 has 1's in  $S_1$  and  $S_4$ , so put

$$SIG_{11} = SIG_{14} = \min\{\infty, h_1(0)\} = 0 + 1 \pmod{5} = 1 \text{ and}$$

$$SIG_{21} = SIG_{24} = \min\{\infty, h_2(0)\} = 3 \cdot 0 + 1 \pmod{5} = 1$$

	$S_1$	$S_2$	$S_3$	$S_4$
$h_1$	1	$\infty$	$\infty$	1
$h_2$	1	$\infty$	$\infty$	1

Signature matrix after considering first row

# COMPUTING SIGNATURE MATRICES IN PRACTICE

- ▶ Consider  $n$  hash functions  
 $h_i : \{0, \dots, m - 1\} \rightarrow$   
 $\{0, \dots, m - 1\}, i = 1, \dots, n$
- ▶ Let  $r$  and  $c$  index rows and columns in the characteristic matrix  $M \in \{0, 1\}^{m \times |C|}$
- ▶ So  $c$  also index columns, while  $i$  indexes rows in the signature matrix  $SIG \in \{1, \dots, m\}^{n \times |C|}$

```
for each  $c$  do
  for  $0 \leq i \leq n$  do
     $SIG(i, c) = \infty$ 
  end for
end for
for each row  $r$  do
  // Iteration 2: second row
  for each column  $c$  do
    if  $M(r, c) = 1$  then
      for  $i=1$  to  $n$  do
         $SIG(i, c) =$ 
           $\min(SIG(i, c), h_i(r))$ 
      end for
    end if
  end for
  // End second row
end for
```

## COMPUTING SIGNATURE MATRICES: EXAMPLE

Row	$S_1$	$S_2$	$S_3$	$S_4$	$x + 1 \pmod{5}$	$3x + 1 \pmod{5}$
0	1	0	0	1	1	1
1	0	0	1	0	2	4
2	0	1	0	1	3	2
3	1	0	1	1	4	0
4	0	0	1	0	0	3

Hash functions computed for a characteristic matrix, with rows indexed from 0 to 4

**Second iteration:** row #1 has 1 in  $S_3$ , so put

$$SIG_{13} = \min\{\infty, h_1(1)\} = 1 + 1 \pmod{5} = 2 \text{ and}$$

$$SIG_{23} = \min\{\infty, h_2(1)\} = 3 + 1 \pmod{5} = 4.$$

	$S_1$	$S_2$	$S_3$	$S_4$
$h_1$	1	$\infty$	2	1
$h_2$	1	$\infty$	4	1

Signature matrix  $M$  after considering second row

# COMPUTING SIGNATURE MATRICES IN PRACTICE

- ▶ Consider  $n$  hash functions  
 $h_i : \{0, \dots, m - 1\} \rightarrow$   
 $\{0, \dots, m - 1\}, i = 1, \dots, n$
- ▶ Let  $r$  and  $c$  index rows and columns in the characteristic matrix  $M \in \{0, 1\}^{m \times |C|}$
- ▶ So  $c$  also index columns, while  $i$  indexes rows in the signature matrix  $SIG \in \{1, \dots, m\}^{n \times |C|}$

```
for each  $c$  do
  for  $0 \leq i \leq n$  do
     $SIG(i, c) = \infty$ 
  end for
end for
for each row  $r$  do
  // Iteration 3: third row
  for each column  $c$  do
    if  $M(r, c) = 1$  then
      for  $i=1$  to  $n$  do
         $SIG(i, c) =$ 
           $\min(SIG(i, c), h_i(r))$ 
      end for
    end if
  end for
  // End third row
end for
```

## COMPUTING SIGNATURE MATRICES: EXAMPLE

<i>Row</i>	$S_1$	$S_2$	$S_3$	$S_4$	$x + 1 \pmod{5}$	$3x + 1 \pmod{5}$
0	1	0	0	1	1	1
1	0	0	1	0	2	4
2	0	1	0	1	3	2
3	1	0	1	1	4	0
4	0	0	1	0	0	3

Hash functions computed for a characteristic matrix, with rows indexed from 0 to 4

**Third iteration:** row # 2 has 1's in  $S_2$  and  $S_4$ , so put

$$SIG_{12} = \min\{\infty, h_1(2)\} = 2 + 1 \pmod{5} = 3,$$

$$SIG_{14} = \min\{SIG_{14}, h_1(2)\} = \min(1, 2 + 1 \pmod{5} = 3) = 1,$$

$$SIG_{22} = \min\{\infty, h_2(2)\} = 6 + 1 \pmod{5} = 2,$$

$$SIG_{24} = \min\{SIG_{24}, h_2(2)\} = \min(1, 6 + 1 \pmod{5} = 2) = 1$$

## COMPUTING SIGNATURE MATRICES: EXAMPLE

<i>Row</i>	$S_1$	$S_2$	$S_3$	$S_4$	$x + 1 \pmod{5}$	$3x + 1 \pmod{5}$
0	1	0	0	1	1	1
1	0	0	1	0	2	4
2	0	1	0	1	3	2
3	1	0	1	1	4	0
4	0	0	1	0	0	3

Hash functions computed for a characteristic matrix, with rows indexed from 0 to 4

	$S_1$	$S_2$	$S_3$	$S_4$
$h_1$	1	3	2	1
$h_2$	1	2	4	1

Signature matrix after considering third row

# COMPUTING SIGNATURE MATRICES IN PRACTICE

- ▶ Consider  $n$  hash functions  
 $h_i : \{0, \dots, m - 1\} \rightarrow$   
 $\{0, \dots, m - 1\}, i = 1, \dots, n$
- ▶ Let  $r$  and  $c$  index rows and columns in the characteristic matrix  $M \in \{0, 1\}^{m \times |C|}$
- ▶ So  $c$  also index columns, while  $i$  indexes rows in the signature matrix  $SIG \in \{1, \dots, m\}^{n \times |C|}$

```
for each  $c$  do
  for  $0 \leq i \leq n$  do
     $SIG(i, c) = \infty$ 
  end for
end for
for each row  $r$  do
  // Iteration 4: fourth row
  for each column  $c$  do
    if  $M(r, c) = 1$  then
      for  $i=1$  to  $n$  do
         $SIG(i, c) =$ 
           $\min(SIG(i, c), h_i(r))$ 
      end for
    end if
  end for
  // End fourth row
end for
```



## COMPUTING SIGNATURE MATRICES: EXAMPLE

<i>Row</i>	$S_1$	$S_2$	$S_3$	$S_4$	$x + 1 \pmod 5$	$3x + 1 \pmod 5$
0	1	0	0	1	1	1
1	0	0	1	0	2	4
2	0	1	0	1	3	2
3	1	0	1	1	4	0
4	0	0	1	0	0	3

Hash functions computed for a characteristic matrix, with rows indexed from 0 to 4

**Fourth iteration:**  $SIG_{11}$  stays 1,  $SIG_{21}$  changes to 0,  $SIG_{13}$  stays 2,  $SIG_{23}$  changes to 0,  $SIG_{14}$  stays 1,  $SIG_{24}$  changes to 0

	$S_1$	$S_2$	$S_3$	$S_4$
$h_1$	1	3	2	1
$h_2$	0	2	0	0

Signature matrix after considering fourth row

# COMPUTING SIGNATURE MATRICES IN PRACTICE

- ▶ Consider  $n$  hash functions  
 $h_i : \{0, \dots, m - 1\} \rightarrow$   
 $\{0, \dots, m - 1\}, i = 1, \dots, n$
- ▶ Let  $r$  and  $c$  index rows and columns in the characteristic matrix  $M \in \{0, 1\}^{m \times |C|}$
- ▶ So  $c$  also index columns, while  $i$  indexes rows in the signature matrix  $SIG \in \{1, \dots, m\}^{n \times |C|}$

```
for each  $c$  do
  for  $0 \leq i \leq n$  do
     $SIG(i, c) = \infty$ 
  end for
end for
for each row  $r$  do
  // Iteration 5: fifth (final) row
  for each column  $c$  do
    if  $M(r, c) = 1$  then
      for  $i=1$  to  $n$  do
         $SIG(i, c) =$ 
           $\min(SIG(i, c), h_i(r))$ 
      end for
    end if
  end for
  // End fifth (final) row
end for
```

# COMPUTING SIGNATURE MATRICES: EXAMPLE

<i>Row</i>	$S_1$	$S_2$	$S_3$	$S_4$	$x + 1 \pmod{5}$	$3x + 1 \pmod{5}$
0	1	0	0	1	1	1
1	0	0	1	0	2	4
2	0	1	0	1	3	2
3	1	0	1	1	4	0
4	0	0	1	0	0	3

Hash functions computed for a characteristic matrix, with rows indexed from 0 to 4

	$S_1$	$S_2$	$S_3$	$S_4$
$h_1$	1	3	0	1
$h_2$	0	2	0	0

Signature matrix after considering fifth row: final signature matrix

# COMPUTING SIGNATURE MATRICES: EXAMPLE

	$S_1$	$S_2$	$S_3$	$S_4$
$h_1$	1	3	0	1
$h_2$	0	2	0	0

Signature matrix after considering fifth row: final signature matrix

- ▶ *Estimates* for Jaccard similarity:  $\text{SIM}(S_1, S_3) = \frac{1}{2}$ ,  $\text{SIM}(S_1, S_4) = 1$
- ▶ *True* Jaccard similarities:  $\text{SIM}(S_1, S_3) = \frac{1}{4}$ ,  $\text{SIM}(S_1, S_4) = \frac{2}{3}$
- ▶ Estimates will be better when raising number of hash functions that is increasing number of rows of the signature matrix

# MINHASHING - ISSUES REMAINING

- ▶ Minhashing is time-consuming, because iterating through all  $m$  rows of  $M$  necessary, and  $m$  is large (huge!)
- ▶ *Thought experiment:*
  - ▶ Imagine using real permutations
  - ▶ Recall: minhash is first row in permuted order with a 1
  - ▶ Consider permutations  $\pi : \{1, \dots, \bar{m}\} \rightarrow \{1, \dots, \bar{m}\}$  for  $\bar{m} < m$
  - ▶ Consider only examining the first  $\bar{m}$  of the permuted rows
  - ▶ Speed up of a factor of  $\frac{m}{\bar{m}}$
  - ▶ However, all first  $\bar{m}$  rows may have 0 in some columns
- ▶ How to deal with that? Can we nevertheless work with only  $\bar{m} < m$  rows and compute sufficiently accurate estimates for the Jaccard similarity of two columns?

# SPEEDING UP MINHASHING: MOTIVATION

- ▶ Continue *thought experiment*
- ▶ Consider computing signature matrices by only examining  $\bar{m} < m$  rows in the characteristic matrix, and using permutations  $\pi : \{1, \dots, \bar{m}\} \rightarrow \{1, \dots, \bar{m}\}$  where
- ▶ the chosen  $\bar{m}$  rows need not be the first  $\bar{m}$  rows
- ▶ For each permutation where no 1 shows, keep  $\infty$  as symbol in the signature matrix  $SIG$
- ▶ *Situation*: Much faster to compute  $SIG$ , but  $SIG(i, c) = \infty$  in some places (how many? is this bad?)
- ▶ Consider computing Jaccard similarities for pairs of columns

# SPEEDING UP MINHASHING: MOTIVATION

## Situation:

- ▶ Compute Jaccard similarities for pairs of columns, while possibly
- ▶  $SIG(i, c) = \infty$  for some  $(i, c)$
- ▶ *Algorithm for estimating Jaccard similarity:*
  - ▶ Row by row, by iterative updates,
  - ▶ maintain count of rows  $a$  where columns agree
  - ▶ maintain count of rows  $d$  where columns disagree
  - ▶ Estimate SIM as  $\frac{a}{a+d}$

## Three cases:

1. *Both columns do not contain  $\infty$  in row:* update counts as usual (either  $a \rightarrow a + 1$  or  $d \rightarrow d + 1$ )
2. *Only one column has  $\infty$  in row:*
  - ▶ Let two rows be  $c_1, c_2$ , and  $SIG(i, c_1) = \infty$ , but  $SIG(i, c_2) \neq \infty$ :
  - ▶ It follows that  $SIG(i, c_1) > SIG(i, c_2)$
  - ▶ So increase count of disagreeing rows by one ( $d \rightarrow d + 1$ )
3. *Both columns have  $\infty$  in a row:* unclear situation, skip updating counts

# SPEEDING UP MINHASHING: MOTIVATION

**Summary:** One determines  $\frac{a}{a+d}$  as estimate for  $SIM(c_1, c_2)$

- ▶ Counts rely on less rows than before. How reliable are they?
- ▶ However, since each permutation only refers to  $\bar{m} < m$  rows, we can afford more permutations
- ▶ The one makes counts less reliable, while the other compensates for it
- ▶ Can we control the corresponding trade-off to our favour?
- ▶ What are the consequences in practice when using real hash functions and not permutations?



## SPEEDING UP MINHASHING: ISSUES TO RESOLVE

- ▶ Let  $T$  be the set of elements of the universal set that correspond to the initial  $\bar{m}$  rows in the characteristic matrix.
- ▶ When executing the above algorithm on only these  $\bar{m}$  rows, we determine

$$\frac{|S_1 \cap S_2 \cap T|}{|(S_1 \cup S_2) \cap T|} \quad (7)$$

as an estimate for the true Jaccard similarity  $\frac{|S_1 \cap S_2|}{|S_1 \cup S_2|}$ .

- ▶ If  $T$  is chosen randomly, the expected value of (7) is the Jaccard similarity  $\text{SIM}(S_1, S_2)$
- ▶ But: there may be some disturbing variation to this estimate

# SPEEDING UP MINHASHING: STRATEGY

## *Idea in practice using hash functions*

- ▶ Divide  $m$  rows into  $\frac{m}{\bar{m}}$  blocks of  $\bar{m}$  rows each
- ▶ For each hash function  $h : \{0, \dots, \bar{m} - 1\} \rightarrow \{0, \dots, \bar{m} - 1\}$ , compute minhash values for each block of  $\bar{m}$  rows
- ▶ Yields  $\frac{m}{\bar{m}}$  minhash values for a single hash function, instead of just one
- ▶ *Extreme:* If  $\frac{m}{\bar{m}}$  is large enough, only one hash function may be necessary
- ▶ *Possible advantage:* By using all  $m$  rows, one balances out errors in the particular estimates on only  $\bar{m}$  of the  $m$  rows:
  - ▶ The overall  $x$  of the type X rows are distributed across blocks of  $\bar{m}$  rows
  - ▶ Likewise, the overall  $y$  type Y rows are distributed across the blocks

## SPEEDING UP MINHASHING: EXAMPLE

$S_1$	$S_2$	$S_3$
0	0	0
0	0	0
0	0	1
0	1	1
1	1	1
1	1	0
1	0	0
0	0	0

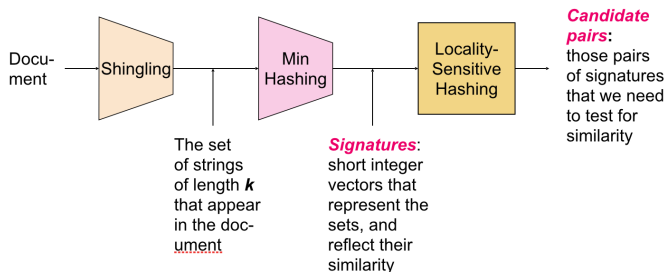
Characteristic matrix for three sets  $S_1, S_2, S_3$ .  $m = 8, \bar{m} = 4$ .

- ▶ *Truth:*  $\text{SIM}(S_1, S_2) = \frac{1}{2}, \text{SIM}(S_1, S_3) = \frac{1}{5}, \text{SIM}(S_2, S_3) = \frac{1}{2}$
- ▶ *Estimate for first four rows:*  
 $\text{SIM}(S_1, S_2) = 0$
- ▶ *Estimate for last four rows:*  
 $\text{SIM}(S_1, S_2) = \frac{2}{3}$  on average across randomly picked hash functions
- ▶ *Overall estimate (expected across randomly picked hash functions):*  $\text{SIM}(S_1, S_2) = \frac{1}{3}$ ,  
Ok estimate for two hash functions

## CURRENT STATUS: ISSUES STILL REMAINING

- ▶ Estimating similarity for each pair of sets may be infeasible even when using minhash signatures just because number of pairs is too large
- ▶ Apart from parallelism nothing can help
- ▶ *Question/Idea:* Can we determine *candidate pairs*, and only compute similarity for them, knowing similarity will be small for all others?
- ▶ *Solution:* *Locality Sensitive Hashing (a.k.a. Near Neighbor Search)*

# SUMMARY OF CURRENT STATUS



From [mmds.org](http://mmds.org)

- ▶ *Shingling*: turning text files into sets ☞ Done!
- ▶ *Minhashing*: computing similarity for large sets ☞ Done!
- ▶ *Locality Sensitive Hashing*: avoids  $O(N^2)$  comparisons by determining candidate pairs ☞ next lecture!

# MATERIALS / OUTLOOK

- ▶ See *Mining of Massive Datasets*, chapter 3.1–3.3
- ▶ As usual, see <http://www.mmds.org/> in general for further resources
- ▶ Next lecture: “Finding Similar Items II”
  - ▶ See *Mining of Massive Datasets* 3.4–3.6