PART 3

BLOCKING/CANOPY GENERATION

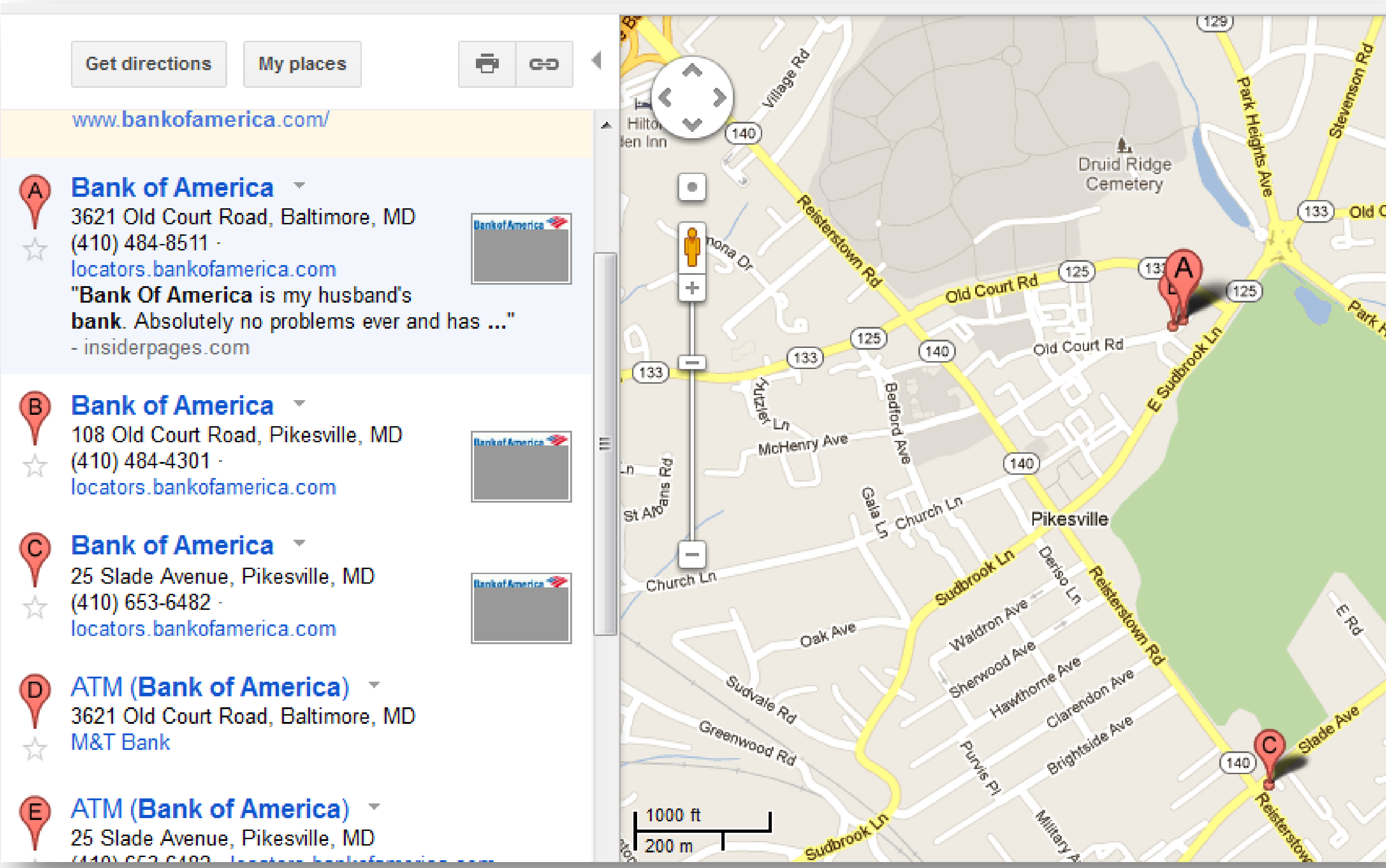
Blocking: Motivation

- Naïve pairwise: $|R|^2$ pairwise comparisons
  - 1000 business listings each from 1,000 different cities across the world
  - 1 trillion comparisons
  - 11.6 days (if each comparison is 1 μs)

- Mentions from different cities are unlikely to be matches
  - 10 million comparisons
  - 10 seconds (if each comparison is 1 μs)

Blocking: Motivation

- Mentions from different cities are unlikely to be matches
  - May miss potential matches



Blocking: Problem Statement

Input: Set of records *R*

Output: Set of *blocks/canopies*

$$\{C_1, C_2, \ldots, C_k\}, where\ \forall_i C_i \subset R\ and \bigcup_i C_i = R$$

Variants:

- *Disjoint Blocking*: Each mention appears in one block.

$$\forall_{i,j} C_i \cap C_j = \emptyset$$

- *Non-disjoint Blocking*: Mentions can appear in more than one block.

Blocking: Problem Statement

$$\{C_1, C_2, \ldots, C_k\}, where\ \forall_i C_i \subset R\ and \bigcup_i C_i = R$$

Metrics:

- Efficiency (or reduction ratio) : $\dfrac{number\ of\ pairs\ compared}{total\ number\ of\ pairs\ in\ R \times R}$

$$= \frac{|\{(x, y) \mid \exists i\ C_i, s.t.\ \ x, y \in C_i\}|}{r(r-1)/2}$$

- Recall* (or pairs completeness) : $\dfrac{number\ of\ true\ matches\ compared}{number\ of\ true\ matches\ in\ R \times R}$

**Need to know ground truth in order to compute this metric*

Blocking: Problem Statement

Metrics:

- Efficiency (or reduction ratio) : $\frac{number\ of\ pairs\ compared}{total\ number\ of\ pairs\ in\ R \times R}$
- Recall* (or pairs completeness) : $\frac{number\ of\ true\ matches\ compared}{number\ of\ true\ matches\ in\ R \times R}$
- Precision* (or pairs quality) : $\frac{number\ of\ true\ matches\ compared}{number\ of\ matches\ compared}$
- Max Canopy Size: $max_i\ |C_i|$

**Need to know ground truth in order to compute this metric*

Blocking Algorithms 1

- Hash based blocking
  - Each block C_i is associated with a hash key h_i.
  - Mention x is hashed to C_i if $hash(x) = h_i$.
  - Within a block, all pairs are compared.
  - Results in disjoint blocks.

- What *hash* function?
  - Deterministic function of attribute values
  - Boolean Functions over attribute values [Bilenko et al ICDM'06, Michelson et al AAAI'06, Das Sarma et al Corr'11]
  - **minHash** (min-wise independent permutations) [Broder et al STOC'98]

Blocking Algorithms 2

- Pairwise Similarity/Neighborhood based blocking
  - Nearby nodes according to a similarity metric are clustered together
  - Results in non-disjoint canopies.

- Techniques
  - Sorted Neighborhood Approach [Hernandez et al SIGMOD'95]
  - Canopy Clustering [McCallum et al KDD'00]

Simple Blocking: Inverted Index on a Key

Examples of blocking keys:

- First three characters of last name
- City + State + Zip
- Character or Token n-grams
- Minimum infrequent n-grams

Learning Optimal Blocking Functions

- Using one or more blocking keys may be insufficient
  - 2,376,206 American’s shared the surname Smith in the 2000 US
  - NULL values may create large blocks.

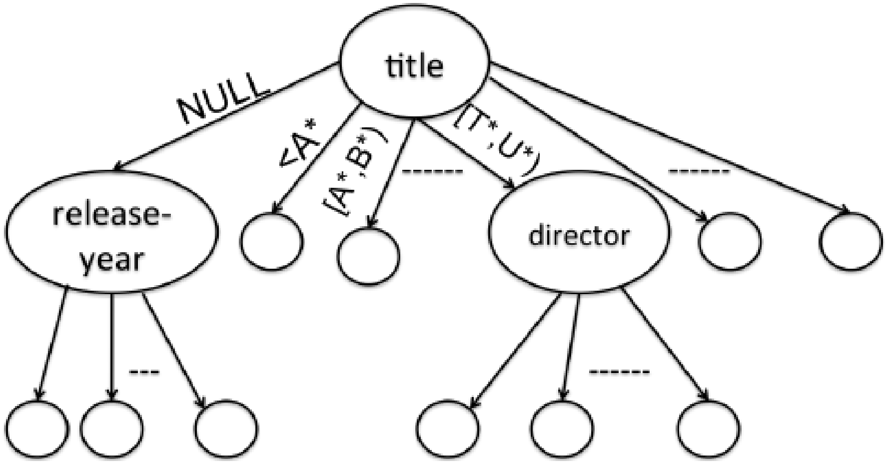
- Solution: Construct blocking functions by combining simple functions

Complex Blocking Functions

- Conjunction of functions [Michelson et al AAAI'06, Bilenko et al ICDM'06]
  - {City} AND {last four digits of phone}

- Chain-trees [Das Sarma et al Corr '11]
  - **If** ({City} = NULL or LA) **then** {last four digits of phone} AND {area code}
    else {last four digits of phone} AND {City}

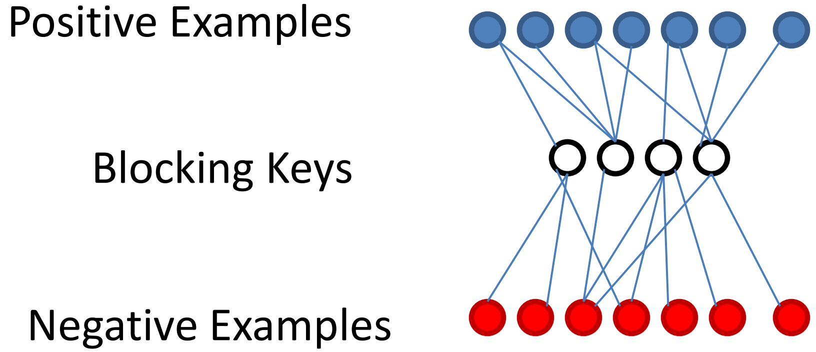
- BlkTrees [Das Sarma et al Corr '11]



Learning an Optimal function [Bilenko et al ICDM '06]

- Find k blocking functions that eliminate the most non-matches, while retaining almost all matches.
  - Need a training set of positive and negative pairs

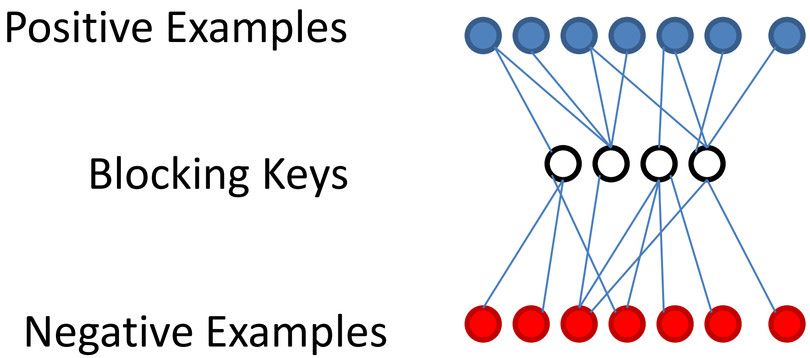
- Algorithm Idea: Red-Blue Set Cover



Pick k Blocking keys such that
(a) At most ε blue nodes are not covered
(b) Number of red nodes covered is minimized

Learning an Optimal function [Bilenko et al ICDM '06]

- Algorithm Idea: Red-Blue Set Cover



Pick k Blocking keys such that
(a) At most ε blue nodes are not covered
(b) Number of red nodes covered is minimized

- Greedy Algorithm:
  - Construct "good" conjunctions of blocking keys $\{p_1, p_2, \ldots\}$.
  - Pick k conjunctions $\{p_{i1}, p_{i2}, \ldots, p_{ik}\}$, such that the following is minimized

$$\frac{number\ of\ new\ blue\ nodes\ covered\ by\ p_{i_j}}{number\ of\ red\ nodes\ covered\ by\ p_{i_j}}$$

minHash (Minwise Independent Permutations)

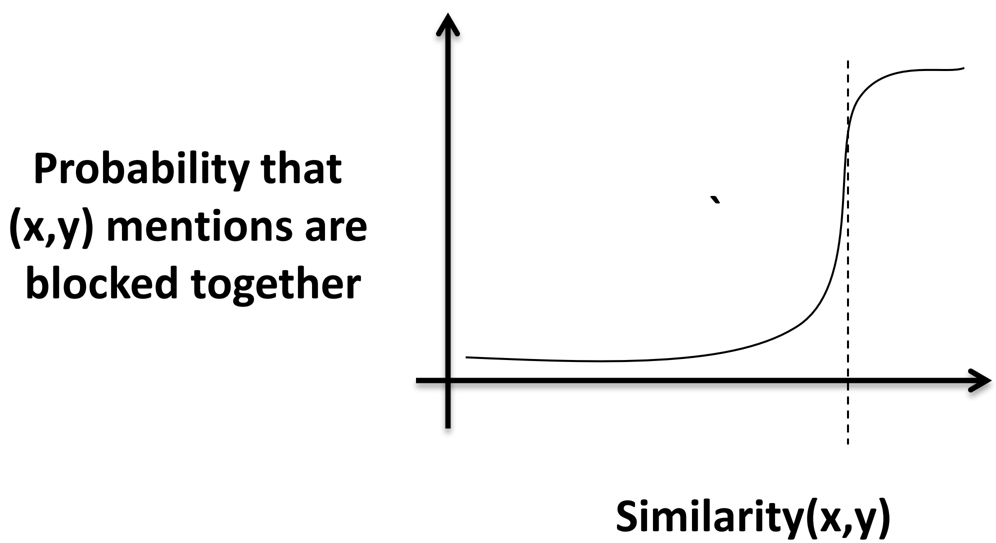
- Let F_x be a set of features for mention x
  - (functions of) attribute values
  - character ngrams
  - optimal blocking functions …
- Let π be a random permutation of features in F_x
  - E.g., order imposed by a random hash function
- *minHash(x)* = minimum element in F_x according to π

Why minHash works?

Surprising property: For a random permutation π,

$$P(minHash(x) = minhash(y)) = \frac{F_x \cap F_y}{F_x \cup F_y}$$

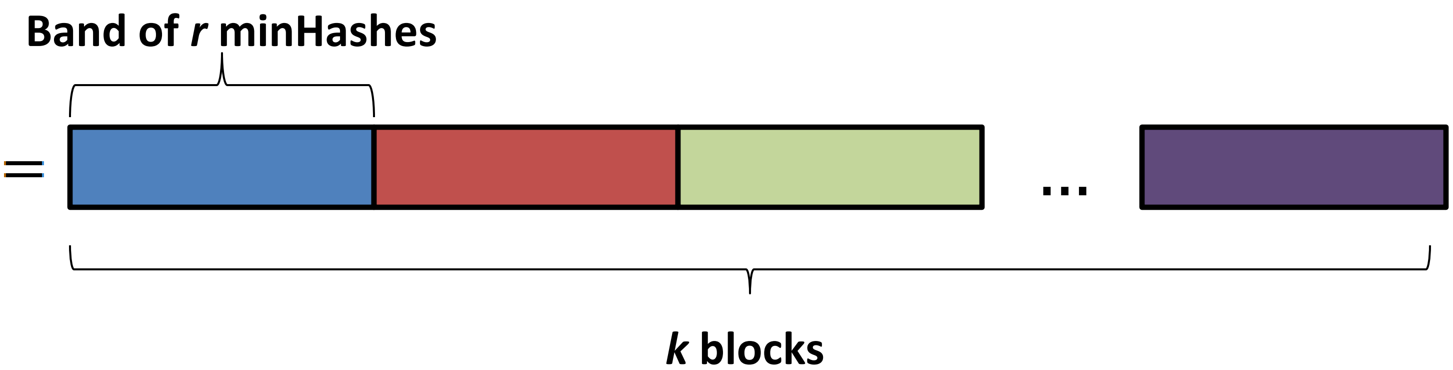
How to build a blocking scheme such that only pairs with Jacquard similarity > s fall in the same block (with high prob)?



Blocking using minHashes

- Compute minHashes using *r* * *k* permutations (hash functions)

$$signature(x) =$$



- Signature's that match on ***1 out of k*** bands, go to the same block.

minHash Analysis

False Negatives: (missing matches)

P(pair x,y not in the same block with Jacquard sim = s) $= (1 - s^r)^k$

should be very low for high similarity pairs

False Positives: (blocking non-matches)

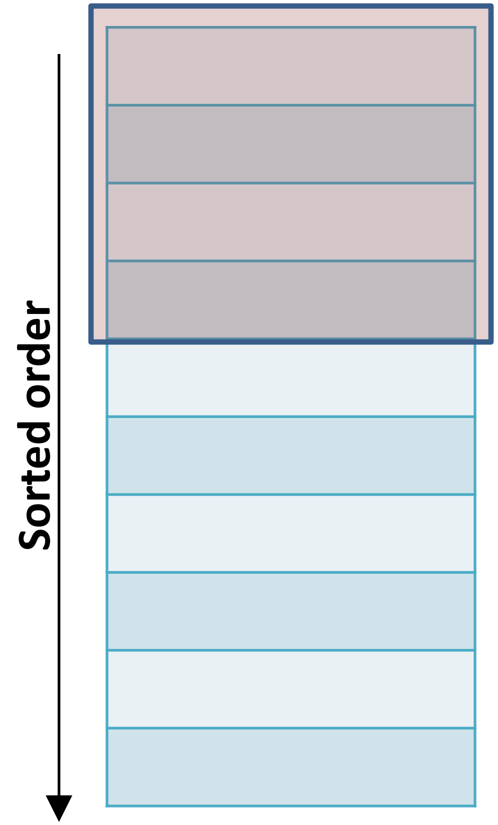
P(pair x,y in the same block with Jacquard sim = s) $= k \times s^r$

$r = 5, k = 20$

Sim(s)	P(not same block)
0.9	10^{-8}
0.8	0.00035
0.7	0.025
0.6	0.2
0.5	0.52
0.4	0.81
0.3	0.95
0.2	0.994
0.1	0.9998

Sorted Neighborhood [Hernandez et al SIGMOD'95]

- Compute a **Key** for each mention.
- **Sort** the mentions based on the key.
- **Merge**: Check whether a record matches with *(w-1)* previous records.
  - Efficient implementation using *Sort Merge Band Join* [DeWitt et al VLDB'91]
- Perform multiple passes with different keys

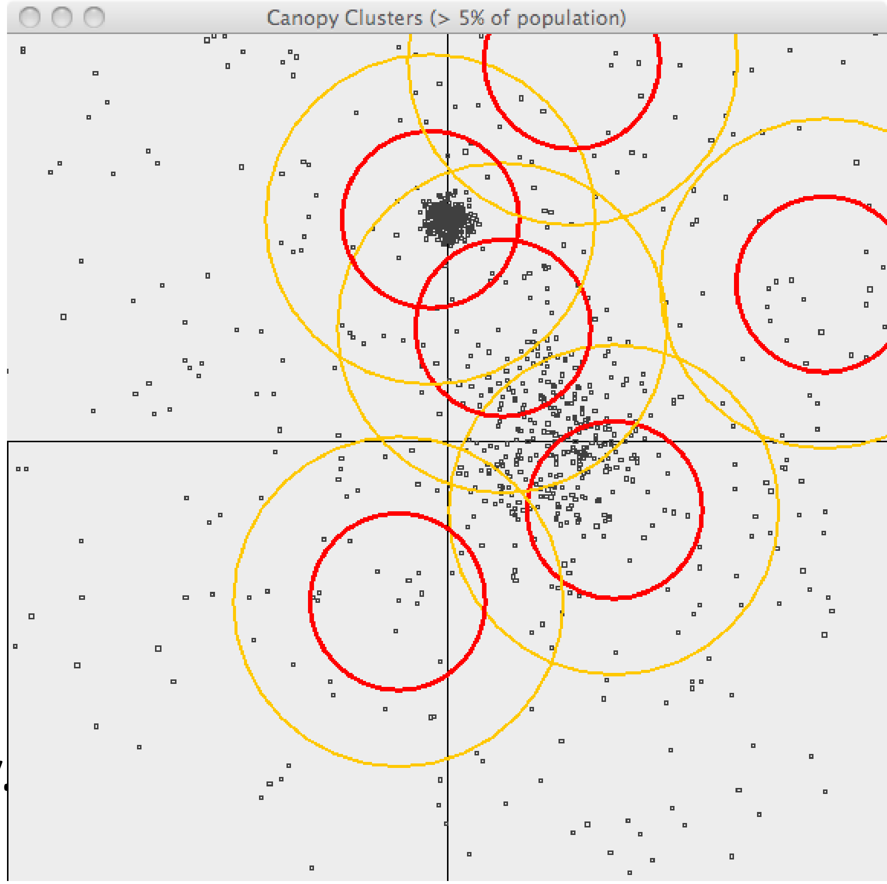


Canopy Clustering [McCallum et al KDD'00]

Input: Mentions M,
$d(x,y)$, a distance metric,
thresholds $T_1 > T_2$

Algorithm:

1. Pick a random element x from M
2. Create new canopy C_x using mentions y s.t. $d(x,y) < T_1$
3. Delete all mentions y from M s.t. $d(x,y) < T_2$
4. Return to Step 1 if M is not empty.



Summary of Blocking

- $O(|R|^2)$ pairwise computations can be prohibitive.
- Blocking eliminates comparisons on a large fraction of non-matches.
- Equality-based Blocking:
  - Construct (one or more) blocking keys from features
  - Records not matching on any key are not compared.
- Similarity based Blocking:
  - Form overlapping canopies of records based on similarity.
  - Only compare records within a cluster.