Поиск дубликатов (Duplicate detection)

Лекция для Computer Science клуба

Александр Уланов / 31 марта 2013

HP Labs Russia

alexander.ulanov@hp.com

Содержание

Поиск дубликатов

- Введение
- Примеры
- Постановка задачи
- Алгоритмы

Введение

Поиск дубликатов

Задача поиска и группировки или согласования различных экземпляров (реализаций) одного и того же объекта

- Различные способы упоминания одного и того же человека в тексте или БД
- Страницы с разными описаниями одних и тех же организаций
- Разные фотографии одних и тех же объектов
- ?

Поиск дубликатов

Поиск дубликатов в поиске дубликатов

Задача имеет много различных названий

Duplicate detection

Coreference resolution

Record linkage

Matching

Reference reconciliation

Object consolidation

Deduplication

Merge/purge

Entity clustering

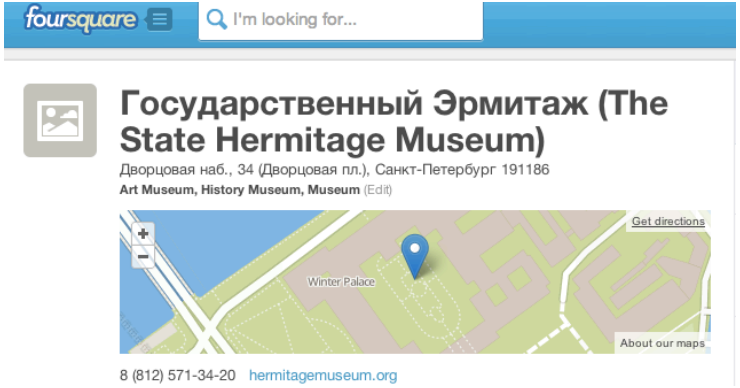
Hardening soft databases

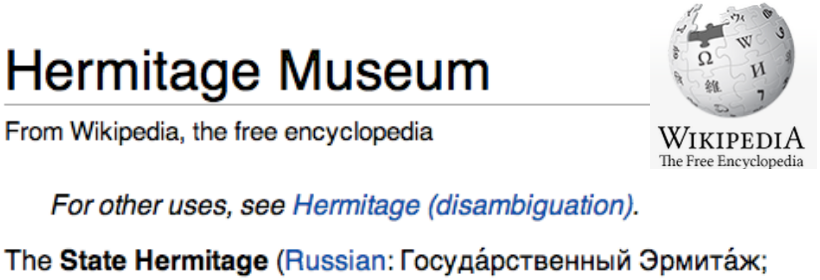
Doubles

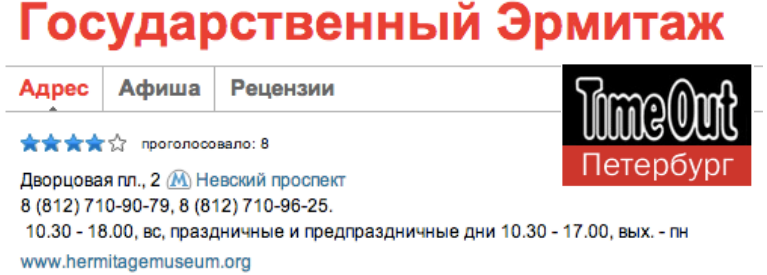
Entity resolution

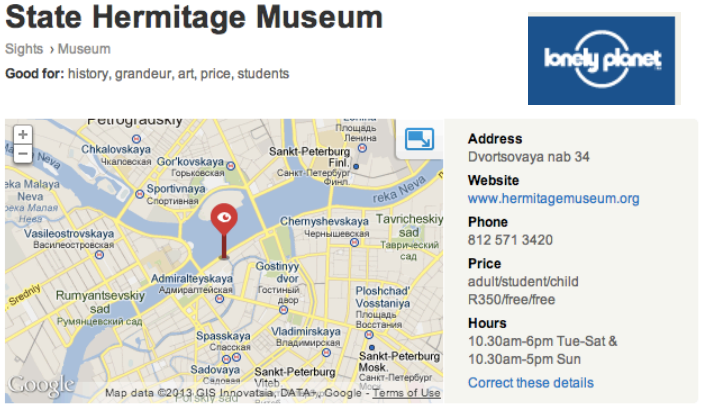
Entity linking

Примеры. Организации



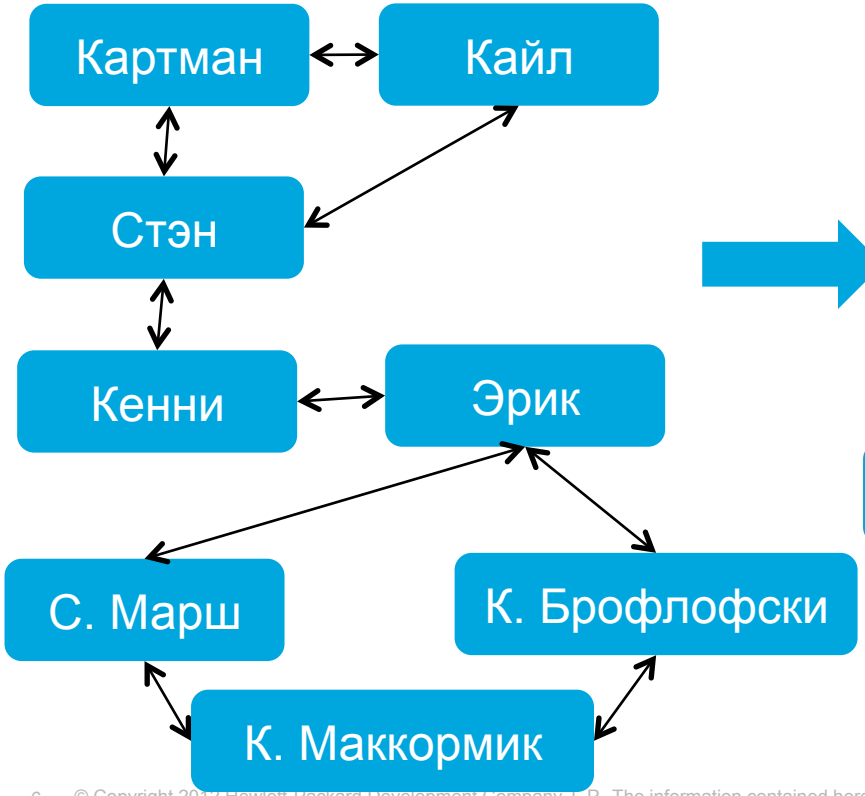




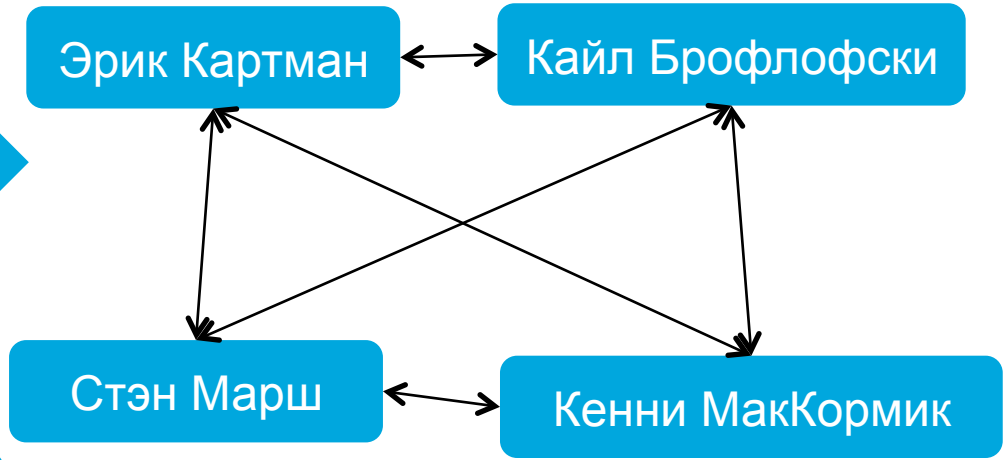


Примеры. Социальная сеть

До поиска дубликатов



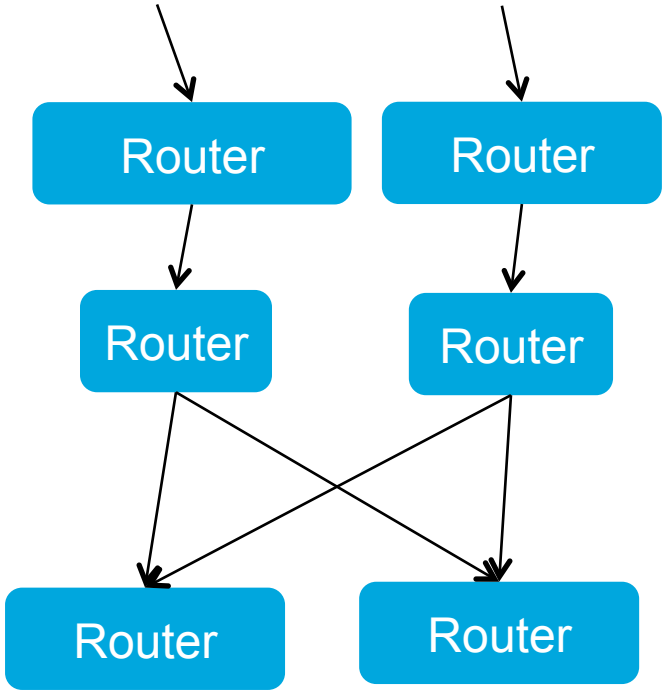
После поиска дубликатов



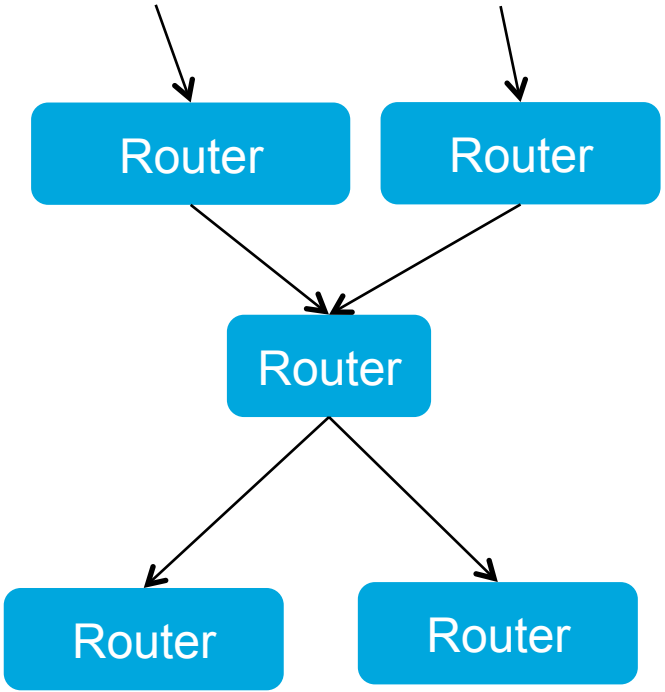
Сеть пригодна для анализа

Примеры. Топология сети Интернет

До поиска дубликатов



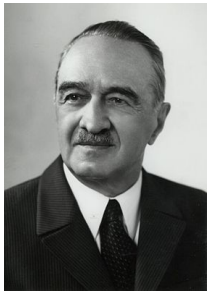
После поиска дубликатов



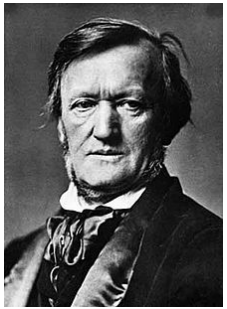
Уменьшена сложность графа

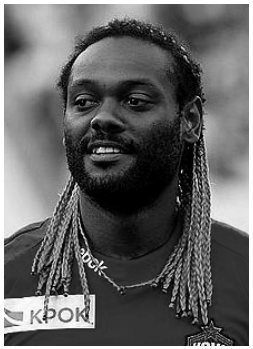
Примеры. Люди

Микоян



Вагнер





*все фото из Википедии

Причины появления дубликатов

Отчего появляются дубликаты?

Основные причины:

- Ошибки ввода
- Пропущенные данные
- Измененные атрибуты
- Формат данных
- Аббревиатуры и сокращения
- Неоднозначность названий и атрибутов объектов

Актуальность

Поиск дубликатов

Основные проблемы

- Большие данные (Big Data)
- Гетерогенность
  - Неструктурированные, «грязные», неполные данные
  - Не просто сравнение фамилий и имен, а сравнение профайла в Фейсбуке со списками твитов в Твиттере и географических координат
- Согласование
  - Не просто дубликаты, а отношения между экземплярами
- Различные приложения
  - БД, социальные сети…

Постановка задачи

Поиск дубликатов

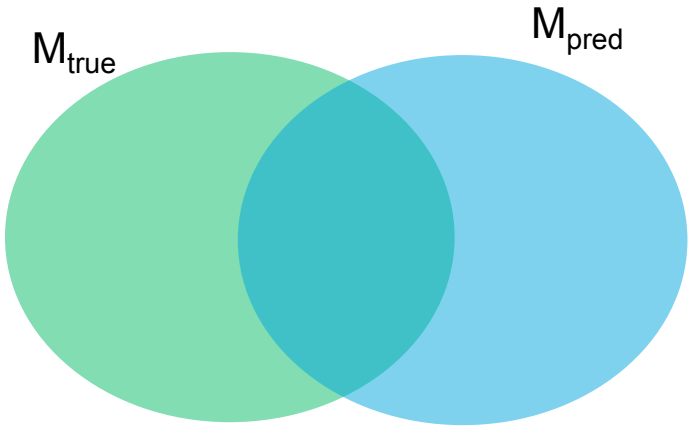
Наиболее типичные задачи

- Поиск дубликатов (duplicate detection)
  - Группировка различных экземпляров одного и того же объекта
- Согласование записей (record linkage, entity linking)
  - Связать записи, которые дублируются в разных источниках
- Ссылки (reference resolution)
  - Сопоставить данные с внешними источниками

Условия

- Каждый экземпляр может быть ассоциирован только с одним объектом
- В задаче согласования записей в одном из источников нет дубликатов
- Если два экземпляра идентичны, то они относятся к одному объекту

Постановка задачи



Обозначения

- R – набор записей (экземпляров)
- H – набор отношений
- M – набор совпадений
- N – набор несовпадений
- E – набор объектов (в реальности)
- L – набор связей (в реальности)

Проверка правильности решения

- Реальные (M_{true}, N_{true}, E_{true}, L_{true}) сравниваются с (M_{pred}, N_{pred}, E_{pred}, L_{pred}), найденными алгоритмом

Метрики

Измерение точности поиска дубликатов

- Попарные (N(N-1)/2)
  - Точность, полнота, F1-мера (precision, recall, F1-measure)

$$P = \frac{tp}{tp + fp} \qquad R = \frac{tp}{tp + fn} \qquad F1 = \frac{PR}{P + R}$$

  - Количество правильно найденных совпадающих пар

- Групповые (кластерные)
  - Чистота (purity)

$$purity(\Omega, C) = \frac{1}{N} \sum_k \max_j \left| \omega_k \cap c_j \right|$$

кластеры

объекты

	Тот же кластер	Другой кластер
Тот же объект	True positive (tp)	False negative (fn)
Другой объект	False positive (fp)	True negative (tn)

Предполагается, что каждый кластер представляет тот объект, реализаций которого в нем больше всего

Решение

Тривиальное решение

Попарное сравнение экземпляров с использованием некоего порога

Как задача классификации

Классифицировать все пары записей в «совпадение» и «не совпадение»

- Несбалансированные данные – «не совпадений» намного больше
- Пары записей не независимы:
  - (A,B)==match, (B,C)==match ➡ (A,C)==match

Как задача кластеризации

Кластеризовать все записи так, чтобы кластеры соответствовали объектам

- В популярных алгоритмах кластеризации предполагается, что количество кластеров *O(R)*
- В задаче поиска дубликатов количество кластеров зачастую близко к R, так как много синглетонов (кластеров с одним элементом)

Алгоритм решения

Обычно включает в себя следующие шаги

- Подготовка данных
  - Нормализация данных
  - Нормализация схемы данных
- Попарное сравнение данных
  - Сравнение атрибутов данных при помощи подходящих функций близости
  - Близость данных – функция от близости их атрибутов
- Применение ограничений
  - Транзитивность
  - Связывание данных
  - Группировка

Подготовка данных

- Нормализация схемы данных
  - Схема – «телефон» и «контактный телефон»
  - Сложные атрибуты - адрес
  - Вложенные атрибуты
  - Наборы атрибутов
  - Сегментация данных
- Нормализация данных
  - Строчные буквы, пробелы
  - Исправление заранее известных ошибок
  - Расшифровка аббревиатур
  - Часто используются словари

Сравнение атрибутов записей

- Для каждой пары записей вычисляются близости их атрибутов
  - sim(A.phone,A.phone),sim(A.name,B.name),…

- **Функции близости (distance functions)**
  - **Булевская**
  - **Редактирования**
    - Levenshtein, Jaro-Winkler, Monge-Elkan
  - **Фонетическая**
    - Soundex
  - **На основе множеств**
    - Dice, Jaccard
  - **На основе векторов**
    - TF-IDF, softTF-IDF
  - **Перевод**
    - Аббревиатуры

SecondString, http://secondstring.sourceforge.net/
Simmetrics: http://sourceforge.net/projects/simmetrics/
LingPipe, http://alias-i.com/lingpipe/index.html

Функция Levenshtein

Вычисляется близость *i* символов из *s* и *j* символов из *t (от начала строк)*

$$D(s,t,i,j) = \min \begin{cases} D(s,t,i-1,j-1) & \text{если } s_i = t_j \\ D(s,t,i-1,j-1)+1 & \text{если производится замена } t_j \text{ на } s_i \\ D(s,t,i,j-1)+1 & \text{если производится вставка } t_j \\ D(s,t,i-1,j)+1 & \text{если производится удаление } s_i \end{cases}$$

Levenshtein. Пример

Levenshtein

Стоимости (берется минимальная из перечисленных)

- Совпадение: если символ совпадает, взять стоимость из левой верхней диагонали
- Замена: если символ не совпадает, взять стоимость из левой верхней диагонали и добавить 1
- Вставка: взять стоимость слева и добавить 1
- Удаление: взять стоимость сверху и добавить 1

		t	h	i	s
	0	1	2	3	4
h	1	1	1	2	3
a	2	2	2	2	3
s	3	3	3	3	**2**

Модификации Levenstein

- Needleman-Wunsch: разные стоимости
- Smith-Waterman: учет начала и конца
- Affine gap: последовательная вставка/ удаление дешевле
- Monge-Elkan: Smith-Waterman+affine

Дистанция

Функция Jaro-Winkler

$$Jaro(s,t) = \frac{1}{3}\left(\frac{|s'|}{|s|} + \frac{|t'|}{|t|} + \frac{|s'| - \tau}{|s'|}\right)$$

$$d = \frac{\max(|s|,|t|)}{2} - 1$$

s', t' – последовательности общих символов

Два символа считаются общими, если $b_j = a_i \quad i - d \le j \le i + d$

τ – половина кол-ва перестановок в последовательности общих символов

Алексей
Алескей

$$d = 2 \quad \tau = 1 \qquad Jaro = \frac{1}{3}\left(\frac{7}{7} + \frac{7}{7} + \frac{7-1}{7}\right) \approx 0.95$$

$$JaroWinkler(s,t) = Jaro(s,t) - i \cdot 0.1 \cdot (1 - Jaro(s,t))$$

i – общий префикс (но не больше 4)

Функции с множествами и векторами

Сравниваемые строки разбиваются на слова

Jaccard, Dice

Отношение пересечения и объединения слов

$$Jaccard(A,B) = \frac{|A \cap B|}{|A \cup B|}$$

$$Dice(A,B) = \frac{2|A \cap B|}{|A| + |B|}$$

TF-IDF

Вычисление косинуса векторов с весами слов

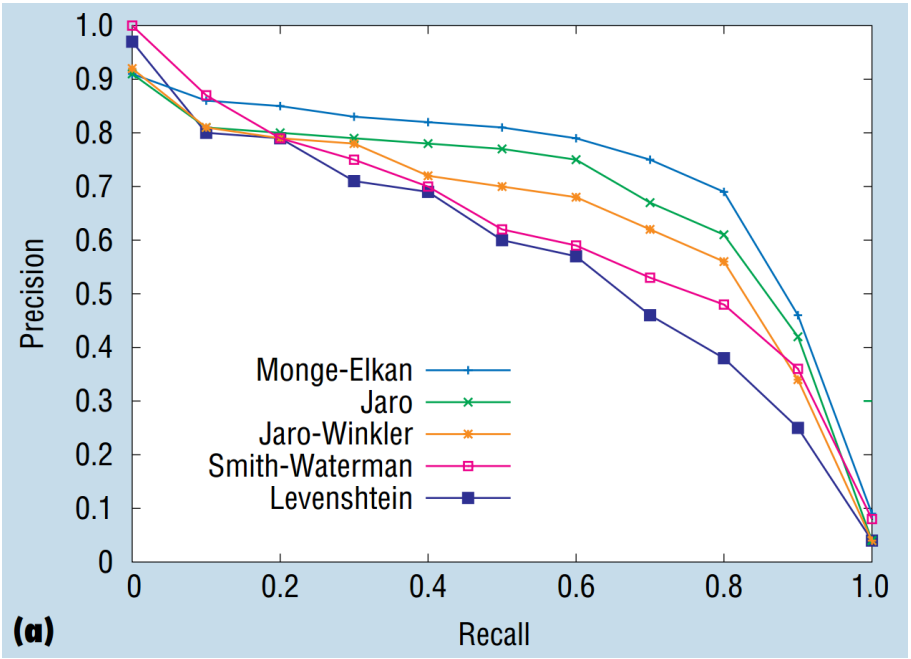
$$Sim_{TFIDF}(S,T) = \sum_{w \in S \cap T} V(w,S) \cdot V(w,T) \quad V(w,S) = \frac{V'(w,S)}{\sqrt{\sum_{w'} V'(w,S)^2}} \quad V'(w,S) = \log(TF_{w,S} + 1) \cdot \log(IDF_w)$$

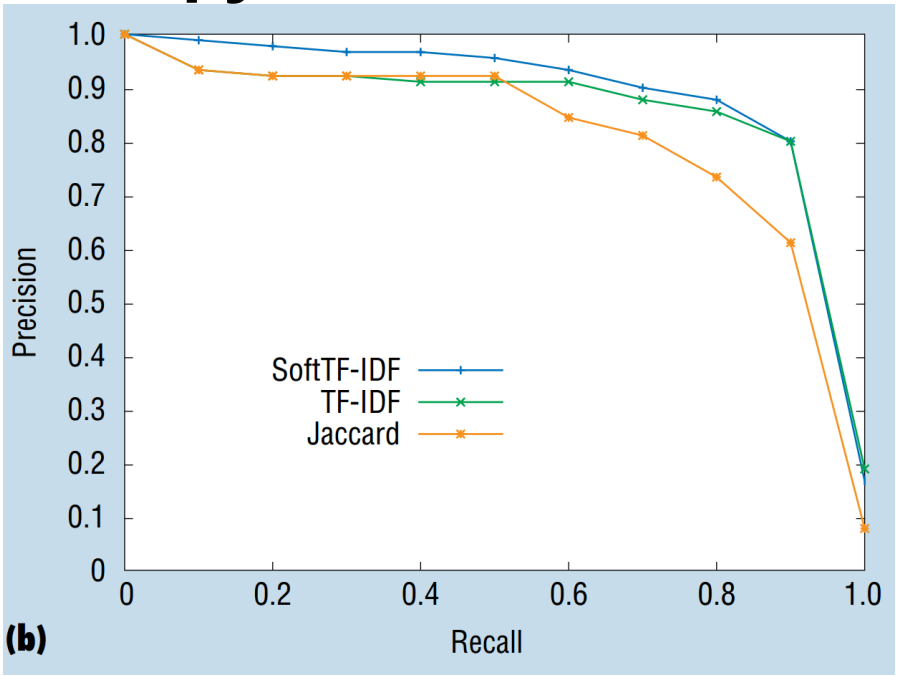
Soft TF-IDF

$$SimSoft_{TFIDF}(S,T) = \sum_{w \in CLOSE(\theta,S,T)} V(w,S) \cdot V(w,T) \cdot N(w,T)$$

$$N(w,T) = \max_{w \in T} sim_{CLOSE} \qquad sim_{CLOSE}(w,v) > \theta$$

Сравнение эффективности функций близости





Средние значения по 11 наборам данных

M. Bilenko, R. Mooney, W. Cohen, P. Ravikumar and S. Fienberg. Adaptive Name Matching in Information Integration. IEEE Intelligent Systems, Vol. 18 Is 5, 2003

Сравнение записей

Задача: вычислить дистанцию между записями, представленными в виде векторов значений функций близости

- Взвешенная сумма (или среднее) дистанций атрибутов
  - Необходимо определять порог
- Правила
  - Необходимо писать правила
- Машинное обучение
  - Необходим набор данных для обучения

Сравнение эффективности дистанций

String metric	Field aggregation method	Maximum F1	Average precision
SoftTF-IDF	SVM	0.792	0.830
	AVG	0.803	0.810
	Single string	0.685	0.782
Jaro	SVM	0.917	0.932
	AVG	0.897	0.922
	Single string	0.728	0.789
Levenshtein	SVM	0.890	0.928
	AVG	0.870	0.920
	Single string	0.865	0.925

Использование машинного обучения и структуры записей

Применение ограничений

Виды ограничений

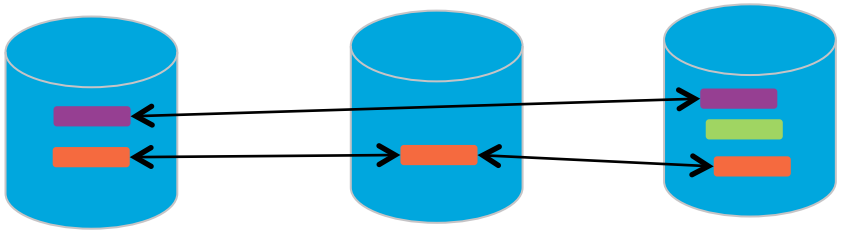
- Транзитивность (для поиска дубликатов)
  - Если M1 и M2, а также M2 и M3 совпадают, то M1 и M3 тоже совпадают
  - Следствие: если M1 и M2, а также M2 и M3 не совпадают, то M1 и M3 *не могут* совпадать
- Эксклюзивность (для согласования данных)
  - Если M1 и M2 совпадают, то M2 и M3 не могут совпадать
  - Следствие: M3 *может* совпадать с какой-либо другой записью M4
- Функциональная зависимость (для очистки данных)
  - Если M1 и M2 совпадают, то M3 и M4 тоже совпадают
  - Следствие: как в транзитивности
- В зависимости от данных

Согласование данных

Согласование

Дано: попарные расстояния между записями в N источниках. Найти: совпадения с максимальной суммой весов

- NP-полная задача, используются субоптимальные алгоритмы на основе попарно несмежных ребер в графе

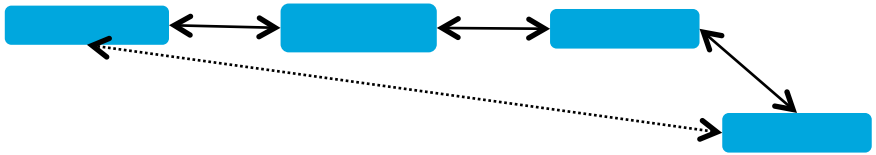


Группировка дубликатов

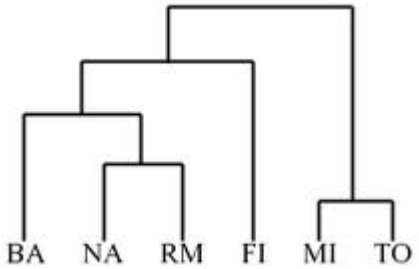
Дубликаты

Дано: попарные расстояния между записями. Найти: группы записей, представляющих один и тот же объект

- Попарное сравнение может дать неполноценный результат (не выполняется транзитивность)
- Иногда транзитивность может быть вредной



- Кластеризация
  - Иерархическая [Bilenko et al, ICDM 05]
  - К-ближайших соседей [Chaudhuri et al, ICDE 05]
- Проблема транзитивности остается
- Сложность определения шага остановки кластеризации



Масштабируемость

Попарных сравнений может быть очень много

Пример

- Население СПб $5*10^6$ человек, сравнений будет ~$12*10^{12}$. Если одно сравнение занимает 1 мс, то весь процесс займет несколько сотен лет

Возможное решение

- Предварительная разбивка на наборы данных, между которыми сравнение не будет произведено [McCallum et al., SIGKDD 2000]
- Простая дистанция

Method	F1	Error	Precision	Recall	Minutes
Canopies	0.838	0.75%	0.735	0.976	7.65
Complete, Expensive	0.835	0.76%	0.737	0.965	134.09
Existing Cora	0.784	1.03%	0.673	0.939	0.03
Author/Year Baseline	0.697	1.60%	0.559	0.926	0.03
Naive Baseline	—	1.99%	1.000	0.000	—

Заключение

Поиск дубликатов

- Очень важен этап нормализации данных
- Выбор функции близости

Проблемы поиска дубликатов

- Большие данные (Big Data)
- Гетерогенность данных
- Масштабируемость

Актуальные задачи

- Согласование аккаунтов в социальных сетях (Twitter, Facebook, Foursquare)
- Согласование данных из различных справочников (Wikipedia, Google Places, Lonely Planet)

Ссылки

Библиотеки

- SecondString, http://secondstring.sourceforge.net/
- Simmetrics: http://sourceforge.net/projects/simmetrics/
- LingPipe, http://alias-i.com/lingpipe/index.html

Литература

- M. Bilenko, R. Mooney, W. Cohen, P. Ravikumar and S. Fienberg. Adaptive Name Matching in Information Integration. *IEEE Intelligent Systems*, Vol. 18 Is 5, 2003 *(практические рекомендации)*
- L. Getoor, A. Machanavajjhala. VLDB 2012 Tutorial on Entity Resolution
- Elmagarmid et al., Duplicate Record Detection: A Survey. *IEEE Transactions on Knowledge and Data Engineering*, Vol. 19, No. 1. (January 2007) *(все, что надо знать о поиске дубликатов на 16 страницах текста)*
- P. Christen. Data Matching Springer 2012 *(большая книга)*

Конец