A Probabilistic Deduplication, Record Linkage and Geocoding System

Peter Christen\(^1\) and Tim Churches\(^2\)

\(^1\) Data Mining Group, Australian National University
\(^2\) Centre for Epidemiology and Research, New South Wales Department of Health

Contact: peter.christen@anu.edu.au

Funded by the ANU, the NSW Department of Health, the Australian Research Council (ARC), and the Australian Partnership for Advanced Computing (APAC)
Outline

- Data cleaning and standardisation
- Record linkage / data integration
- *Febrl* overview
- Probabilistic data cleaning and standardisation
- Blocking / indexing
- Record pair classification
- Parallelisation in *Febrl*
- Data set generation
- Geocoding
- Outlook
Data cleaning and standardisation (1)

Real world data is often *dirty*
- Missing values, inconsistencies
- Typographical and other errors
- Different coding schemes / formats
- Out-of-date data

Names and addresses are especially prone to data entry errors

Cleaned and standardised data is needed for
- Loading into databases and data warehouses
- Data mining and other data analysis studies
- Record linkage and data integration
Data cleaning and standardisation (2)

Remove unwanted characters and words
Expand abbreviations and correct misspellings
Segment data into well defined output fields
The task of linking together records representing the same entity from one or more data sources. If no unique identifier is available, probabilistic linkage techniques have to be applied.

Applications of record linkage:
- Remove duplicates in a data set (internal linkage)
- Merge new records into a larger master data set
- Create customer or patient oriented statistics
- Compile data for longitudinal studies
- Geocode data

*Data cleaning and standardisation are important first steps for successful record linkage*
Record linkage techniques

- Deterministic or exact linkage
  - A *unique identifier* is needed, which is of high quality (precise, robust, stable over time, highly available)
  - For example *Medicare, ABN* or *Tax file* number (are they *really* unique, stable, trustworthy?)

- Probabilistic linkage (*Fellegi & Sunter, 1969*)
  - Apply linkage using available (personal) information
  - Examples: *names, addresses, dates of birth*

- Other techniques
  (rule-based, fuzzy approach, information retrieval)
Febrl – Freely extensible biomedical record linkage

- An experimental platform for new and improved linkage algorithms
- Modules for data cleaning and standardisation, record linkage, deduplication and geocoding
- Open source [https://sourceforge.net/projects/febrl/](https://sourceforge.net/projects/febrl/)
- Implemented in *Python* [http://www.python.org](http://www.python.org)
  - Easy and rapid prototype software development
  - Object-oriented and cross-platform (*Unix, Win, Mac*)
  - Can handle large data sets stable and efficiently
  - Many external modules, easy to extend
Three step approach

1. Cleaning
   – Based on look-up tables and correction lists
   – Remove unwanted characters and words
   – Correct various misspellings and abbreviations

2. Tagging
   – Split input into a list of words, numbers and separators
   – Assign one or more tags to each element of this list
     (using look-up tables and some hard-coded rules)

3. Segmenting
   – Use either rules or a hidden Markov model (HMM)
     to assign list elements to output fields
**Hidden Markov model (HMM)**

- A HMM is a *probabilistic* finite state machine
  - Made of a set of *states* and *transition probabilities* between these states
  - In each state an *observation* symbol is emitted with a certain probability distribution
  - In our approach, the observation symbols are *tags* and the states correspond to the *output fields*
For an observation sequence we are interested in the most likely path through a given HMM (in our case an observation sequence is a tag list).

The Viterbi algorithm is used for this task (a dynamic programming approach).

Smoothing is applied to account for unseen data (assign small probabilities for unseen observation symbols).
Probabilistic data cleaning and standardisation – Example

Uncleaned input string: ’Doc. peter Paul MILLER’
Cleaned into string: ’dr peter paul miller’

Word and tag lists:
['dr', 'peter', 'paul', 'miller']
['TI', 'GM/SN', 'GM', 'SN']

Two example paths through HMM
1: Start -> Title (TI) -> Givenname (GM) -> Middlename (GM) -> Surname (SN) -> End
2: Start -> Title (TI) -> Surname (SN) -> Givenname (GM) -> Surname (SN) -> End
Number of possible links equals the product of the sizes of the two data sets to be linked.

Performance bottleneck in a record linkage system is usually the (expensive) evaluation of similarity measures between record pairs.

Blocking / indexing techniques are used to reduce the large amount of record comparisons.

*Febrl* contains (currently) three indexing methods:

- Standard blocking
- Sorted neighbourhood approach
- Fuzzy blocking using $n$-grams (e.g. bigrams)
Record pair classification

For each record pair compared a vector containing matching weights is calculated

Example:
Record A: ['dr', 'peter', 'paul', 'miller']
Record B: ['mr', 'pete', '', 'miller']
Matching weights: [0.2, 0.8, 0.0, 2.4]

Matching weights are used to classify record pairs as links, non-links, or possible links

Fellegi & Sunter classifier simply sums all the weights, then uses two thresholds to classify

Improved classifiers are possible (for example using machine learning techniques)
**Parallelisation**

- Implemented transparently to the user
- Currently using *MPI* via Python module *PyPar*
- Use of super-computing centres is problematic (privacy) → Alternative: *In-house office clusters*
- Some initial performance results (on *Sun SMP*)

![Graphs showing speedup vs number of processors for Step 1 and Step 2]
Data set generation

- Difficult to acquire data for testing and evaluation (as record linkage deals with names and addresses)

- Also, linkage status is often not known (hard to evaluate and test new algorithms)

- *Febrl* contains a data set generator
  - Uses frequency tables for given- and surname, street name and type, suburb, postcode, age, etc.
  - Uses dictionaries of known misspellings
  - *Duplicate records* are created via random introduction of modifications (like insert/delete/transpose characters, swap field values, delete values, etc.)
## Data set generation – Example

<table>
<thead>
<tr>
<th>REC_ID</th>
<th>ADDRESS1</th>
<th>ADDRESS2</th>
<th>SUBURB</th>
</tr>
</thead>
<tbody>
<tr>
<td>rec-0-org</td>
<td>wylly place</td>
<td>pine ret vill</td>
<td>taree</td>
</tr>
<tr>
<td></td>
<td>wylly place</td>
<td>pine ret vill</td>
<td>taree</td>
</tr>
<tr>
<td>rec-0-dup-0</td>
<td></td>
<td>wylly place</td>
<td>taree</td>
</tr>
<tr>
<td>rec-0-dup-1</td>
<td></td>
<td>pine ret vill</td>
<td>taree</td>
</tr>
<tr>
<td>rec-0-dup-2</td>
<td></td>
<td>wylly place</td>
<td>tared</td>
</tr>
<tr>
<td>rec-0-dup-3</td>
<td></td>
<td>wylly parade</td>
<td>taree</td>
</tr>
<tr>
<td>rec-1-org</td>
<td>stuart street</td>
<td>hartford</td>
<td>menton</td>
</tr>
<tr>
<td>rec-2-org</td>
<td>griffiths street</td>
<td>myross</td>
<td>kilda</td>
</tr>
<tr>
<td>rec-2-dup-0</td>
<td></td>
<td>griffith street</td>
<td>myross</td>
</tr>
<tr>
<td>rec-2-dup-1</td>
<td></td>
<td>griffith street</td>
<td>mycross</td>
</tr>
<tr>
<td>rec-3-org</td>
<td>ellenborough place</td>
<td>kalkite homestead</td>
<td>sydney</td>
</tr>
</tbody>
</table>

Each record is given a unique identifier, which allows the evaluation of accuracy and error rates for record linkage.
**Geocoding**

- The process of matching addresses with geographic locations (longitude and latitude)

**Geocoding tasks**
- Preprocess the geocoded reference data (cleaning, standardisation and indexing)
- Clean and standardise the user addresses
- (Fuzzy) match of user addresses with the reference data
- Return location and match status

**Match status:** address, street or locality level

**Geocode reference data used:** G-NAF
**Geocoded national address file**

- **G-NAF**: Available since early 2004
- Source data from 13 organisations
  (around 32 million source records)
- Processed into 22 normalised database tables
Febrl geocoding system

Only NSW G-NAF data available
(around 4 million address, 58,000 street and 5,000 locality records)

Additional Australia Post and GIS data used
(for data imputing and to compute neighbouring regions)
Outlook

- Several research areas
  - Improving probabilistic data standardisation
  - New and improved blocking / indexing methods
  - Apply machine learning techniques for record pair classification
  - Improve performances (scalability and parallelism)
- Project web page

Febri is an ideal experimental platform to develop, implement and evaluate new data standardisation and record linkage algorithms and techniques.