Febrl – A parallel open source record linkage and geocoding system

Peter Christen

Data Mining Group, Australian National University
in collaboration with
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)

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  - Out-of-date data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

Record linkage and data integration

- 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

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
- Free, open source https://sourceforge.net/projects/febrl/
- Implemented in Python 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

Data cleaning and standardisation (2)

Name

Address

Date of Birth

Doc Type Name Unit

Title Geography Suburb

Street Locality

Title Month Year

record

name

mobile

42Main Rd App. C.T. 2600

26/4/1986

Remove unwanted characters and words
Expand abbreviations and correct misspellings
Segment data into well defined output fields

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)

Probabilistic data cleaning and standardisation

Three step approach in Febrl

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
**Step 1: Cleaning**

- Assume the input component is one string (either name or address – dates are processed differently)
- Convert all letters into lower case
- Use correction lists which contain pairs of original:replacement strings
- An empty replacement string results in removing the original string
- Correction lists are stored in text files and can be modified by the user
- Different correction lists for names and addresses

**Step 2: Tagging**

- Cleaned strings are split at whitespace boundaries into lists of words, numbers, characters, etc.
- Using look-up tables and some hard-coded rules, each element is tagged with one or more tags

Example:
- Uncleaned input string: “Doc. peter Paul MILLER”
- Cleaned string: “dr.peter paul miller”

Word and tag lists:
- [‘dr’, ‘peter’, ‘paul’, ‘miller’]

**Step 3: Segmenting**

- Using the tag list, assign elements in the word list to the appropriate output fields
- Rules based approach (e.g. AutoStar)
  - Example: “if an element has tag ‘TI’ then assign the corresponding word to the ‘Title’ output field”
  - Hard to develop and maintain rules
  - Different sets of rules needed for different data sets
- Hidden Markov model (HMM) approach
  - A machine learning technique (supervised learning)
  - Training data is needed to build HMMs

**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

**HMM probability matrices**

<table>
<thead>
<tr>
<th>Observation</th>
<th>Start</th>
<th>Title</th>
<th>Givenname</th>
<th>Middlename</th>
<th>Surname</th>
<th>End</th>
</tr>
</thead>
<tbody>
<tr>
<td>TI</td>
<td>96%</td>
<td>1%</td>
<td>1%</td>
<td>1%</td>
<td>–</td>
<td>–</td>
</tr>
<tr>
<td>GM</td>
<td>1%</td>
<td>35%</td>
<td>33%</td>
<td>15%</td>
<td>–</td>
<td>–</td>
</tr>
<tr>
<td>GF</td>
<td>1%</td>
<td>35%</td>
<td>27%</td>
<td>14%</td>
<td>–</td>
<td>–</td>
</tr>
<tr>
<td>SN</td>
<td>1%</td>
<td>9%</td>
<td>14%</td>
<td>45%</td>
<td>–</td>
<td>–</td>
</tr>
<tr>
<td>UN</td>
<td>1%</td>
<td>20%</td>
<td>25%</td>
<td>25%</td>
<td>–</td>
<td>–</td>
</tr>
</tbody>
</table>

**HMM data segmentation**

- 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)

**Address HMM standardisation example**

- Raw input: ‘73 Miller St, NORTH SYDENY 2060’
  - Cleaned into: ‘73 miller St, north sydney 2060’
- Word and tag lists:

- Example path through HMM
  - Start -> Wayfare Number (NU) -> Wayfare Name (NM) -> Wayfare Type (WT) -> Locality Name (LN) -> Postcode (PC) -> End
**Address standardisation results**

- Various NSW Health data sets
  - HMM1 trained on 1,450 Death Certificate records
  - HMM2 contains HMM1 plus 1,000 Midwifes Data Collection training records
  - HMM3 is HMM2 plus 60 unusual training records
- AutoStan rules (for ISC) developed over years

<table>
<thead>
<tr>
<th>Test Data Set</th>
<th>HMM</th>
<th>HMM</th>
<th>AutoStan</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>(1,000 records each)</td>
<td>1</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>Death Certificates</td>
<td>95.7%</td>
<td>96.8%</td>
<td>97.6%</td>
</tr>
<tr>
<td>Inpatient Statistics Collection</td>
<td>95.7%</td>
<td>95.9%</td>
<td>97.4%</td>
</tr>
</tbody>
</table>

**Blocking / indexing**

- 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) comparison of field values (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', 'john', '', 'miller']
  - Matching weights: [0.2, -3.2, 0.6, 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)

**HMM training (1)**

- Both transition and observation probabilities need to be trained using training data (maximum likelihood estimates (MLE) are derived by accumulating frequency counts for transitions and observations)
- Training data consists of records, each being a sequence of taq:hmm_state pairs
- Example (2 training records):
  - #`42 / 131 miller place manly 2095 new_south_wales`
    - [0.2, -3.2, 0.6, 2.4]
  - #`2 richard street lewisham 2049 new_south_wales`
    - [0.2, -3.2, 0.6, 2.4]

**HMM training (2)**

- A bootstrapping approach is applied for semi-automatic training
  1. Manually edit a small number of training records and train a first rough HMM
  2. Use this first HMM to segment and tag a larger number of training records
  3. Manually check a second set of training records, then train an improved HMM
- Only a few person days are needed to get a HMM that results in an accurate standardisation (instead of weeks or even months to develop rules)

**Name standardisation results**

- NSW Midwifes Data Collection (1990 - 2000)
  - (around 963,000 records, no medical information)
  - 10-fold cross-validation study with 10,000 random records (9,000 training and 1,000 test records)
- Both Febrl rule based and HMM data cleaning and standardisation
- Rules were better because most names were simple (not much structure to learn for HMM)

**Field comparison functions in Febrl**

- Exact string
- Truncated string (only consider beginning of strings)
- Approximate string (using Winkler, Edit dist, Bigram etc.)
- Encoded string (using Soundex, NYSIS, etc.)
- Keying difference (allow a number of different characters)
- Numeric percentage (allowing percentage tolerance)
- Numeric absolute (allow absolute tolerance)
- Date (allow day tolerance)
- Age (allow percentage tolerance)
- Time (allow minute tolerance)
- Distance (allow kilometre tolerance)

**Final linkage decision (F & S)**

- The final weight is the sum of weights of all fields
- Record pairs with a weight above an upper threshold are designated as a link
- Record pairs with a weight below a lower threshold are designated as a non-link
- Record pairs with a weight between are possible link
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)

Data set generation – Example

- Data set with 4 original and 6 duplicate records

```
REC_ID  ADDRESS1  ADDRESS2  SUBURB
rec-0-org,  wylly place,  pine ret vill,  taree
rec-0-dup-1,  pine ret vill,  wylly place,  taree
rec-0-dup-2,  wylly place,  pine ret vill,  taree
rec-0-dup-3,  wylly place,  pine ret vill,  taree
rec-1-org,  stuart street,  hartford,  menton
rec-2-org,  griffiths street,  myross,  kilda
rec-2-dup-0,  griffiths street,  myross,  kilda
rec-2-dup-1,  griffiths street,  myross,  kilda
rec-3-org,  ellenborough place,  kalkite homestead,  sydney
```

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

Geocoding techniques

- Street centreline based (many commercial systems)
- Property parcel centre based (our approach)
- A recent study found substantial differences (especially in rural areas)
  
  Cayo and Talbot; Int. Journal of Health Geographics, 2003

Geocoding system

```
G-NAF data

G-NAF data

Locality file

GIS data

Address data

Street data

Identities

Geocoding system

Data

Webservice servers

Data

```

- 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

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 surnames, street names and types, suburbs, postcodes, etc.
  - Duplicate records are created via random introduction of modifications (like insert/delete/transpose characters, swap field values, delete values, etc.)
  - Also uses lists of known misspellings

Geocoded national address file

```
G-NAF: Available since early 2004

Sourcedata from 13 organisations

Processed into 22 normalised database tables
```

Additional data files

- Use external Australia Post postcode and suburb look-up tables for correcting and imputing (e.g., if a suburb has a unique postcode this value can be imputed if missing, or corrected if wrong)
- Use boundary files for postcodes and suburbs to build neighbouring region lists
  
  Idea: People often record neighbouring suburb or postcode if it has a higher perceived social status

  Create lists for direct and indirect neighbours (neighbouring levels 1 and 2)
**Febrl geocoding match engine**

- Uses cleaned and standardised user address(es) and G-NAF inverted index data
- Fuzzy rule based approach
  1. Find street match set (street name, type and number)
  2. Find postcode and locality match set (with no, then direct, then indirect neighbour levels)
  3. Intersect postcode and locality sets with street match set (if no match increase neighbour level and go back to 2.)
  4. Refine with unit, property, and building match sets
  5. Retrieve corresponding location (or locations)
  6. Return location and match status (address, street or locality level match; none, one or many matches)

**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

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

**Contributions / Acknowledgements**

- Dr Tim Churches (New South Wales Health Department, Centre for Epidemiology and Research)
- Dr Markus Hegland (ANU Mathematical Sciences Institute)
- Dr Lee Taylor (New South Wales Health Department, Centre for Epidemiology and Research)
- Ms Kim Lim (New South Wales Health Department, Centre for Epidemiology and Research)
- Mr Alan Willmore (New South Wales Health Department, Centre for Epidemiology and Research)
- Mr Karl Goiser (ANU Computer Science PhD student)
- Mr Justin Zhu (ANU Computer Science honours student, 2002)
- Mr David Horgan (ANU Computer Science summer student, 2003/2004)
- Mr Puthick Hok (ANU Computer Science honours student, 2004)