A Two-Step Classification Approach to Unsupervised Record Linkage

Peter Christen

Department of Computer Science, The Australian National University
Canberra ACT 0200, Australia
Email: peter.christen@anu.edu.au

Abstract

Linking or matching databases is becoming increasingly important in many data mining projects, as linked data can contain information that is not available otherwise, or that would be too expensive to collect manually. A main challenge when linking large databases is the classification of the compared record pairs into matches and non-matches. In traditional record linkage, classification thresholds have to be set either manually or using an EM-based approach. More recently developed classification methods are mainly based on supervised machine learning techniques and thus require training data, which is often not available in real world situations or has to be prepared manually. In this paper, a novel two-step approach to record pair classification is presented. In a first step, example training data of high quality is generated automatically, and then used in a second step to train a supervised classifier. Initial experimental results on both real and synthetic data show that this approach can outperform traditional unsupervised clustering, and even achieve linkage quality almost as good as fully supervised techniques.

Keywords: Data linkage, data matching, deduplication, entity resolution, clustering, support vector machines, quality measures.

1 Introduction

With many businesses, government organisations and research projects collecting large amounts of data, techniques that allow efficient processing, analysing and mining of massive databases have in recent years attracted interest from both academia and industry. Increasingly, data from various sources has to be linked, matched and aggregated in order to improve data quality, or to enrich existing data with additional information. Similarly, detecting and removing duplicate records that relate to the same entity within one database is often required in the data pre-processing step of many data mining projects. The aim of such linkages and deduplications is to match and aggregate all records that relate to the same entity, such as a patient, a customer, a business, a product description, a publication, or a genome sequence.

Record or data linkage and deduplication can be used to improve data quality and integrity (Winkler 2004), to allow re-use of existing data sources for new studies, and to reduce costs and efforts in data acquisition. In the health sector, for example, linked data might contain information that is needed to improve health policies (Kelman et al. 2002), and that traditionally has been collected with time consuming and expensive survey methods. Statistical agencies routinely link census data for further analysis (Gill 2001), while businesses often deduplicate their databases to compile mailing lists or link them for collaborative e-Commerce projects. Within taxation offices and departments of social security, record linkage is used to identify people who register for assistance multiple times or who work and collect unemployment benefits. Another application of current interest is the use of record linkage in crime and terror detection. Security agencies and crime investigators increasingly rely on the ability to quickly access files for a particular individual (Wang et al. 2006), which may help to prevent crimes and terror by early intervention.

The problem of finding similar entities does not only apply to records that refer to persons. In bioinformatics, record linkage can help finding genome sequences in large data collections that are similar to a new, unknown sequence at hand. Increasingly important is the removal of duplicates in the results returned by Web search engines and automatic text indexing systems, where copies of documents (for example bibliographic citations) have to be identified and filtered out before being presented to the user (Bhattacharya and Getoor 2007). Finding and comparing consumer products from several online stores is another application of growing interest (Bilenko et al. 2005). As product descriptions are often slightly different, matching them becomes difficult.

If unique entity identifiers (or keys) are available in all databases to be linked, then the problem of linking at the entity level becomes trivial: a simple database *join* is all that is required. However, in most cases no unique keys are shared by all databases, and more sophisticated linkage techniques need to be applied. These techniques can be broadly classified into deterministic, probabilistic, and modern approaches (Christen and Goiser 2007, Winkler 2006).

A general schematic outline of the record linkage process is given in Figure 1. As most real-world data collections contain noisy, incomplete and incorrectly formatted information, data cleaning and standardisation are important pre-processing steps for successful record linkage, and also before data can be loaded into data warehouses or used for further mining (Rahm and Do 2000). A lack of good quality data can be one of the biggest obstacles to successful record linkage and deduplication (Clarke 2004). The main task of data cleaning and standardisation is the conversion of the raw input data into well defined, consistent forms, as well as the resolution of inconsistencies in the way information is represented and encoded (Churches et al. 2002).

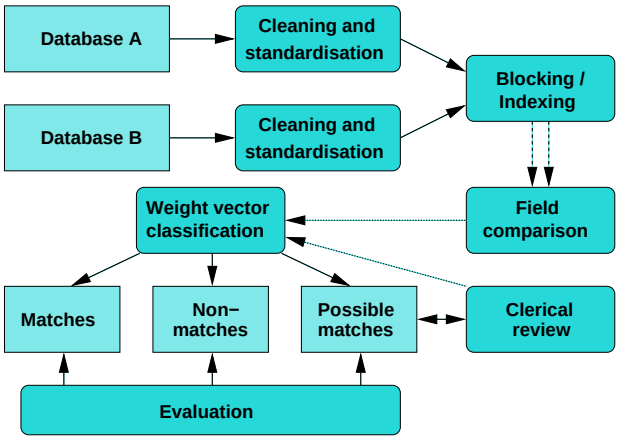

Figure 1: General record linkage process. The output of the blocking step are candidate record pairs, while the comparison step produces weight vectors with numerical similarity weights.

R1:	Christine	Smith	42	Main	Street
R2:	Christina	Smith	42	Main	St
R3:	Bob	O'Brian	11	Smith	Rd
R4:	Robert	Bryce	12	Smythe	Road

WV(R1,R2):	0.9	1.0	1.0	1.0	0.9
WV(R1,R3):	0.0	0.0	0.0	0.0	0.0
WV(R1,R4):	0.0	0.0	0.5	0.0	0.0
WV(R2,R3):	0.0	0.0	0.0	0.0	0.0
WV(R2,R4):	0.0	0.0	0.5	0.0	0.0
WV(R3,R4):	0.7	0.4	0.5	0.7	0.9

Figure 2: Four example records (made of given name and surname; and street number, name and type attributes) and the corresponding weight vectors resulting from the comparisons of these records.

If two databases, **A** and **B**, are to be linked, potentially each record from **A** has to be compared with all records from **B**. The total number of potential record pair comparisons thus equals the product of the size of the two databases, $|\mathbf{A}| \times |\mathbf{B}|$, with $|\cdot|$ denoting the number of records in a database. Similarly, when deduplicating a database, **A**, the total number of potential record pair comparisons is $|\mathbf{A}| \times (|\mathbf{A}|-1)/2$, as each record potentially has to be compared to all others. The performance bottleneck in a record linkage or deduplication system is usually the expensive detailed comparison of fields (or attributes) between pairs of records (Baxter et al. 2003, Christen and Goiser 2007), making it unfeasible to compare all pairs when the databases are large. Assuming there are no duplicate records in the databases (i.e. one record in database **A** can only match to one record in database **B**, and vice versa), then the maximum number of true matches corresponds to the number of records in the smaller database. Therefore, while the computational efforts increase quadratically, the number of potential true matches only increases linearly when linking larger databases. This also holds for deduplication, where the number of duplicate records is always less than the number of records in a database.

To reduce the large amount of potential record pair comparisons, record linkage methods employ some form of indexing or filtering techniques, collectively known as *blocking* (Baxter et al. 2003): a single record attribute or a combination of attributes, often called the *blocking key*, is used to split the databases into blocks. All records that have the same value in the blocking key will be inserted into one block, and candidate record pairs are then generated only from records within the same block. These candidate pairs are compared using a variety of comparison functions applied to one or more (or a combination of) record attributes. These functions can be as simple as an exact string or a numerical comparison, can take variations and typographical errors into account (Cohen et al. 2003, Christen 2006), or can be as complex as a distance comparison based on look-up tables of geographic locations (longitudes and latitudes).

Each comparison returns a numerical similarity value (called *matching weight*), often in normalised form. Two attribute values that are equal, therefore, would have a similarity of 1, while the similarity of two completely different values would be 0. Attribute values that are somewhat similar would have a similarity value somewhere between 0 and 1. As illustrated in Figure 2, a vector (called *weight vector*) is formed for each compared record pair containing all the matching weights calculated by the different comparison functions. These weight vectors are then used to classify record pairs into *matches*, *non-matches*, and *possible matches*, depending upon the decision model used (Christen and Goiser 2007, Fellegi and Sunter 1969, Gu and Baxter 2006). Record pairs that were removed by the blocking process are classified as non-matches without being compared explicitly.

Two records that have the same values in all their attributes will with high likelihood refer to the same entity, as it is very unlikely that two entities have the same values in all their attributes. The weight vector calculated when comparing such a pair of records will have matching weights of 1 in all vector elements. On the other hand, weight vectors that have 0 or very low similarity values in all their elements are with high likelihood the result of a comparison of two records that refer to different entities, as it is highly unlikely that two records that refer to the same entity have different values in all their record attributes. For example, even if a woman changes her surname and her address when she gets married, her date of birth and her maiden name will stay the same.

From this follows that it is often easy to classify with high accuracy record pairs that are very similar as matches, and pairs that are very dissimilar as non-matches. On the other hand, it is much more difficult to classify pairs that have some similar and some dissimilar attribute values. This is illustrated in Figure 2, where records *R1* and *R2* are very similar, with only two minor difference in the given name and street type attributes (which usually are taken care of in the data cleaning and standardisation step (Churches et al. 2002)), and thus very likely refer to the same person. On the other hand, records *R3* and *R4* are more different to each other, and it is not obvious if they refer to the same person.

Based on the above observations, it is possible to automatically extract training examples (weight vectors) from the set of all weight vectors that with high likelihood correspond to true matches or true non-matches, and to then use these weight vectors to train a supervised classifier. From the six weight vectors shown in Figure 2, *WV(R1,R2)* can be used as a training example for matches, while *WV(R1,R3)* and *WV(R2,R3)*, and possibly even *WV(R1,R4)* and *WV(R2,R4)*, can be used for non-matches.

This two-step approach to automated record pair classification, which has been inspired by similar approaches that were developed for text classification (Basu et al. 2002, Liu et al. 2003, Nigam et al. 2000, Yu et al. 2002), is presented in more detail in Section 3, and evaluated experimentally in Section 4. First, in the following section, an overview of related research is presented. Conclusions and an outlook to future work is then given in Section 5.

2 Related Work

The classical probabilistic record linkage approach, as developed by Fellegi and Sunter (1969), has been improved in recent years mainly through application of the expectation-maximisation (EM) algorithm for better parameter estimation in record pair classification (Winkler 2000), and through the use of approximate string comparisons to calculate partial agreement weights when attribute values have typographical variations (Christen 2006, Winkler 2006).

In the late 1990s researchers started to explore the use of techniques originating in machine learning, data mining, artificial intelligence, information retrieval and database research to improve the linkage process. Many of these approaches are based on supervised learning techniques and assume that training data is available (i.e. record pairs with known true match and true non-match status). However, such training examples are often not available in real world situations, or have to be prepared manually (an expensive and time consuming process).

One supervised approach is to learn distance measures for approximate string comparisons, such as the costs for character inserts, deletes and substitutions for edit-distance (Bilenko and Mooney 2003, Cohen et al. 2003), with the aim to adapt similarity computations to a particular data domain. Decision tree induction (Elfeky et al. 2002, Neiling 2005, Tejada et al. 2002) and support vector machines (SVM) (Nahm et al. 2002) are two popular supervised machine learning techniques that have been employed successfully for record pair classification. These techniques usually achieve better linkage quality compared to unsupervised methods.

In (Elfeky et al. 2002), three approaches to record pair classification are described; the first based on supervised decision trees, the second using unsupervised k-means clustering (with three clusters, one each for matches, possible matches and non-matches), and the third being a hybrid approach that combines the first two to overcome the problem of lack of training data. In this hybrid approach, a sub-set of weight vectors is clustered in a first step (again into matches, possible matches and non-matches), and the match and non-match clusters are then used as training data for a supervised classifier in a second step. Both the fully supervised and hybrid approach outperformed the clustering approach in experimental studies.

Active learning is another approach, aimed at reducing the amount of training data required. In (Sarawagi and Bhamidipaty 2002), a system is described that presents a difficult to classify record pair to a user for manual classification. After such a pair is classified manually, it is added to the training set and the classifiers are re-trained. This process is repeated until all record pairs are successfully classified. The authors reported that manually classifying less than 100 training pairs using their approach provided better results than a fully supervised approach that used 7,000 randomly selected examples. A similar approach has been presented in (Tejada et al. 2002), where a committee of decision trees is used to learn a set of rules that describe linkages.

Unsupervised clustering techniques have been investigated both for improved blocking (Cohen and Richman 2002, McCallum et al. 2000) and for automatic record pair classification (Elfeky et al. 2002). The k-means clustering algorithm has been used in (Gu and Baxter 2006) to group weight vectors into matches and non-matches (i.e. $k = 2$). In this approach, a user can identify a 'fuzzy' region in the middle between the two cluster centroids where the difficult to classify record pairs are located. These pairs will then be given to the user for manual clerical review. Using synthetic data, it was shown that this approach can significantly reduce the number of record pairs that have to be reviewed manually, while keeping high linkage quality. In (Goiser and Christen 2006), the clustering techniques k-means and farthest-first were compared with supervised decision tree induction on both synthetic and real data sets. Surprisingly, the simple farthest-first technique achieved results comparable to decision trees.

Another area where unsupervised techniques have been explored in recent years is entity resolution of relational data based on relational clustering (Bhattacharya and Getoor 2007). While the techniques described so far assume that only similarities between attribute values of record pairs are available for classification, in relational data the entities have additional relational information that can be used to improve the quality of entity resolution. Relational information includes, for example, census databases that contain a family relationship attribute (with values such as 'married to', 'dependent of', or 'parent of'); or bibliographic data where, besides the name of a paper, a publication record also contains a list of authors. Two author names in different publications that have several co-authors in common in other publications will more likely refer to the same real person compared to an author with the same name that has different co-authors. Experimental results (Bhattacharya and Getoor 2007) on various data sets have shown that collective relational entity resolution outperforms non-relational entity resolution that is based on record pair similarities only. However, there are still many situations in the real world where no relational data is available, and this paper concentrates on improving the unsupervised classification of such non-relational data.

The two-step approach presented here has been inspired by similar approaches to text classification, where often only a small number of labeled positive examples and a very large number of unlabeled examples are available. The aim is then to learn a binary classifier from these positive and unlabeled examples. In (Yu et al. 2002), the PEBL approach is presented, which is based on iteratively training a SVM using the positive and a selected set of strong negative examples. More unlabeled examples are included into the negative training set as the trained classifier becomes more accurate, until all unlabeled examples are classified. A comparison of different approaches to learning from positive and unlabeled examples is given in (Liu et al. 2003). The techniques compared were PEBL, Naïve Bayes classification, Rocchio text classification in combination with SVM, and an EM based approach (called S-EM) that uses 'spy' documents, positive examples that are inserted into the set of unlabeled documents to better model their distributions (Liu et al. 2002). A new approach, that uses a biased SVM formulation, is then proposed that achieved better classification results than all previous methods (Liu et al. 2003).

In a related text classification scenario, only small numbers of both positive and negative labeled training examples, as well as a large number of unlabeled examples, are available. In (Nigam et al. 2000), a combination of the EM and Naïve Bayes classifiers is presented. Training is started using only the labeled data, and then iteratively refined using the unlabeled examples. The experimental results presented showed that this approach was able to reduce classification errors by up to 30%.

Also related to the work presented here is semi-supervised clustering (Basu et al. 2002), which is based on the idea of using a small amount of labeled data to initialise the cluster centroids, for example for k-means, rather than using random centroid ini-

tialisation. Experimental results discussed in (Basu et al. 2002) show that this can significantly improve cluster quality. In the area of record linkage, such an approach can be taken for classifying weight vectors, by initialising two cluster centroids, one to the exact similarity values (matches) and the other to total dissimilarity values (non-matches). Such a clustering approach will be compared to other classification techniques in Section 4 below.

3 Two-step Record Pair Classification

The idea behind the approach presented in this paper is based on the following two assumptions. First, the weight vectors generated in the comparison step that have exact or high similarity values in all their vector elements were with high likelihood produced when two records were compared that refer to the same entity, as it is very unlikely that two different entities have high similarities in all their attributes. Second, weight vectors with mostly low similarity values were with high likelihood produced when two records were compared that refer to different entities, as it is highly unlikely that two records that refer to the same entity have different values in all their attributes.

Thus, the hypothesis investigated in this paper is that it is possibly to select in a first step weight vectors as training examples that with high likelihood correspond to either true matches or true non-matches, and to then use these examples in a second step to train a supervised classifier. This paper concentrates on the first step, and presents and evaluates several approaches to automatically select training examples. Combined, these two steps will allow fully automated, unsupervised record pair classification, without the need to know the true match and non-match status of the weight vectors produced in the comparison step.

3.1 Step 1: Selection of Training Examples

There are two main approaches to selecting training examples, either using thresholds or nearest-based. As illustrated in Figure 1, pairs of records that were generated in the blocking step are compared using d comparison functions (with $d \geq 1$), resulting in a set $\mathbf{W}$ of weight vectors $\mathbf{w}_i$ ($1 \leq i \leq |\mathbf{W}|$) of length d containing matching weights (similarity values), with $|\cdot|$ denoting the number of elements in a set. It is assumed that all comparison functions return normalised similarity values between 1.0 (exact similarity) and 0.0 (total dissimilarity), i.e. $0.0 \leq \mathbf{w}_i[j] \leq 1.0, 1 \leq j \leq d, \forall \mathbf{w}_i \in \mathbf{W}$. The weight vector that contains exact similarities in all its vector elements is denoted by $\mathbf{m}$ (i.e. $\mathbf{m}[j] = 1.0, 1 \leq j \leq d$), and the weight vector that contains total dissimilarities only by $\mathbf{n}$ (i.e. $\mathbf{n}[j] = 0.0, 1 \leq j \leq d$).

The aim of the training example selection process is to choose weight vectors from $\mathbf{W}$ that with very high likelihood correspond to true matches and true non-matches, respectively, and to insert them into two sets, the match example training set, $\mathbf{W}_M$, and the non-match example training set, $\mathbf{W}_N$. Generally, only a fraction of all weight vectors will be selected for training, and thus it is expected that $(|\mathbf{W}_M| + |\mathbf{W}_N|) \ll |\mathbf{W}|$. In the following, the two approaches to training example selection are presented in more detail.

3.1.1 Threshold-based Selection

In this approach, one threshold for matches, t_m (with $0.0 < t_m < 1.0$), and one for non-matches, t_n (with $0.0 < t_n < 1.0$), are used to select weight vectors that have all their similarity values either within t_m of the exact match value (1.0) or within t_n of the total dissimilarity value (0.0). More formally, the match and non-match example sets $\mathbf{W}_M$ and $\mathbf{W}_N$ are formed according to:

$$\begin{aligned} \mathbf{W}_M &= \{\mathbf{w}_i \in \mathbf{W} : (\mathbf{m}[j] - \mathbf{w}_i[j]) \leq t_m, 1 \leq j \leq d\}, \\ \mathbf{W}_N &= \{\mathbf{w}_i \in \mathbf{W} : (\mathbf{n}[j] + \mathbf{w}_i[j]) \leq t_n, 1 \leq j \leq d\}. \end{aligned}$$

Depending upon the values of t_m and t_n, there is the possibility that a weight vector could be included into both training example sets $\mathbf{W}_M$ and $\mathbf{W}_N$. In such a situation, this weight vector will be removed from both $\mathbf{W}_M$ and $\mathbf{W}_N$, as it cannot be a good quality training example for both matches and non-matches. For example, this would happen when $t_m = t_n = 0.6$ for a weight vector which has all similarity values set to 0.5, i.e. $\mathbf{w}_i[j] = 0.5, 1 \leq j \leq d$.

3.1.2 Nearest-based Selection

Rather than using thresholds, in this approach the weight vectors closest to $\mathbf{m}$ are selected into $\mathbf{W}_M$, and the weight vectors closest to $\mathbf{n}$ into $\mathbf{W}_N$. More formally, if x_m and x_n (with $x_m > 0$ and $x_n > 0$) are the number of weight vectors to be selected into $\mathbf{W}_M$ and $\mathbf{W}_N$, respectively, and the distance between two weight vectors is calculated using the Manhattan distance as $dist(\mathbf{w}_i, \mathbf{w}_k) = \sum_{j=1}^{d} |\mathbf{w}_i[j] - \mathbf{w}_k[j]|$, then the training example sets are formed according to:

$$\begin{aligned} \mathbf{W}_M &= \{\mathbf{w}_i \in \mathbf{W}, \mathbf{w}_k \notin \mathbf{W}_M : dist(\mathbf{m}, \mathbf{w}_i) < dist(\mathbf{m}, \mathbf{w}_k)\}, \\ \mathbf{W}_N &= \{\mathbf{w}_i \in \mathbf{W}, \mathbf{w}_k \notin \mathbf{W}_N : dist(\mathbf{w}_i, \mathbf{n}) < dist(\mathbf{w}_k, \mathbf{n})\}, \end{aligned}$$

with $x_m = |\mathbf{W}_M|$ and $x_n = |\mathbf{W}_N|$.

There are two variations of how the x_m and x_n nearest vectors can be chosen. First, they can be selected regardless if some of them contain the same values in all of their vector elements. For example, there might be a number of weight vectors that contain only exact match values (i.e. that are equal to $\mathbf{m}$) if there are pairs of records that are exact matches, i.e. that have the same values in all compared attributes. Similarly, as illustrated in Figure 2, there will be a large number of weight vectors that only contain total dissimilarity values (i.e. weight vectors that are equal to $\mathbf{n}$). In the worst case, the weight vectors selected into $\mathbf{W}_M$ will all be equal to $\mathbf{m}$ and the weight vectors selected into $\mathbf{W}_N$ will all be equal to $\mathbf{n}$. This situation would not be very useful for training the classifier in step two. Thus, in order to make sure weight vectors with different values are selected, the x_m and x_n nearest *unique* vectors can be inserted into the sets $\mathbf{W}_M$ and $\mathbf{W}_N$ of training examples. These two variations will be referred to as *non-unique* and *unique* nearest in the experimental results presented in Section 4 below.

A second variation in the nearest-based approach is how to choose the values of x_m and x_n. Both can be set to the same value, resulting in a balanced classification problem that has the same number of match and non-match training examples. However, as discussed in Section 1 earlier, the number of true non-matches in the set of weight vectors generated by the blocking and comparison steps will likely be much larger than the number of true matches, because the number of true matches is usually limited by the size of the smaller data set. Classifying the weight vector set $\mathbf{W}$ is therefore an imbalanced classification problem, and this should be reflected in the number of training examples provided to the classifier in step

Data set	Number of records	Task	Pairs completeness	Reduction ratio	Number of weight vectors (i.e. $\|\mathbf{W}\|$)	Ratio of true matches to true non-matches
Census	449 + 392	Linkage	1.000	0.988	2,093	1 / 5.40
Restaurant	864	Deduplication	1.000	0.713	106,875	1 / 953.24
DS-Gen	1,000	Deduplication	0.957	0.995	2,475	1.13 / 1
DS-Gen	2,500	Deduplication	0.940	0.997	9,878	1 / 2.06
DS-Gen	5,000	Deduplication	0.953	0.997	35,491	1 / 4.48
DS-Gen	10,000	Deduplication	0.948	0.997	132,532	1 / 9.32

Table 1: Data sets used in experiments. See Section 4.1 for more details.

two. An estimation of the ratio of matches to non-matches, r, can be calculated based on the number of records in both data sets, $|\mathbf{A}|$ and $|\mathbf{B}|$, and the number of weight vectors $|\mathbf{W}|$:

$$r = \frac{min(|\mathbf{A}|, |\mathbf{B}|)}{|\mathbf{W}| - min(|\mathbf{A}|, |\mathbf{B}|)}. \quad (1)$$

The number of weight vectors selected into the match examples training set $\mathbf{W}_M$ will therefore usually be smaller than the number of vectors selected into the non-match examples training set $\mathbf{W}_N$. In the experiments presented in Section 4 below, the results for this variation will be shown in two separate tables.

3.2 Step 2: Classification of Record Pairs

Once example training data for matches, $\mathbf{W}_M$, and non-matches, $\mathbf{W}_N$, has been selected, any binary classifier can be trained on them, followed by the classification of the weight vectors that have not been selected as training examples, i.e. $\mathbf{W}_T = \mathbf{W} \setminus (\mathbf{W}_M \cup \mathbf{W}_N)$. In this paper, a support vector machine (SVM) classifier will be used, as this technique can handle high-dimensional data and has shown to be robust to noisy data. The use of other classifiers, such as decision trees, is possible and will be investigated as part of future work.

One important issue that is also left for future work is that the example training data generated automatically in the first step will be linearly separable, as the two training sets only contain examples that are either close to the exact match value or close to the total dissimilarity value. Thus, there will be a 'gap' between the match and non-match training examples. Similar to the inclusion of 'spy' documents in the S-EM approach (Liu et al. 2002), adding randomly sampled weight vectors from this 'gap' into the training example sets should improve the overall classification accuracy. This idea is currently being implemented and results will be reported elsewhere.

4 Experimental Evaluation

In this section, the different approaches to automatically select training examples for matches and non-matches will be compared with three other classification methods. The first is a linear kernel SVM that uses all weight vectors and their match status for supervised classification (10-fold cross validation results are reported). The second is the standard k-means clustering approach using Euclidean distance and with two clusters (one for the matches and one for the non-matches), with the cluster centroids initialised to the exact match vector $\mathbf{m}$ and total dissimilarity vector $\mathbf{n}$, respectively. The third is an 'optimal threshold' classifier that has access to the match status of all weight vectors, and that emulates an optimal probabilistic approach (Fellegi and Sunter 1969). It sums each weight vector into a single matching weight (i.e. it generates 1-dimensional weight vectors), and then finds the optimal classification threshold using these matching weights that minimises the number of false matches and false non-matches.

All techniques described here were implemented in the *Febrl* (Christen et al. 2004) open source record linkage system,[1] which is written in the Python programming language. For SVM classification the *PyML*[2] Python module was used, which is based on the *libsvm* library (Chang and Lin 2001). The default linear kernel SVM method from *PyML* was chosen in all experiments presented here. Further experiments using SVMs with non-linear kernels and other classification approaches are planned for future work. All reported experiments were carried out on a Dell Optiplex GX280 with an Intel Pentium 3 GHz CPU and 2 Gigabytes of main memory, running Linux 2.6.20 (Ubuntu 7.04 Feisty Fawn) and using Python 2.5.1.

4.1 Data Sets and Linkage Setup

The proposed approaches were evaluated using both real and synthetic data sets, which are summarised in Table 1. Two small real data sets were taken from the *SecondString* toolkit,[3] while artificial data sets of various sizes were created using the *Febrl* data set generator (Christen 2005). This generator works by first creating a number of *original* records based on frequency tables containing real world names (given- and surname) and addresses (street number, name and type; postcode; suburb and state name), followed by the random generation of *duplicates* of these records based on modifications (like inserting, deleting or substituting characters, and swapping, removing, inserting, splitting or merging words), also based on real error characteristics. All data sets generated for this paper contained 60% original and 40% duplicate records, with up to nine duplicates for one original record (the number of duplicates created per original record are 'Zipf' distributed), and with a maximum of three modifications per attribute and maximum ten modifications per record.

A standard blocking approach (Baxter et al. 2003) was used for all experiments reported here, with the blocking keys being combinations of name, address and postcode values. For the record pair comparison step, the Winkler (Christen 2006, Winkler 2004) approximate string comparator (commonly employed in record linkage for name comparisons) was used on the name and address attributes. Additionally, for the Census and synthetic data sets, character difference comparisons were used on the zipcode, postcode, street number and state abbreviation attributes.

The pairs completeness measure shown in Table 1 is the number of true matched record pairs generated by a blocking technique divided by the total number of true matched pairs (Christen and Goiser 2007). It measures how effective a blocking technique is in generating true matched record pairs. Pairs completeness

[1] http://febrl.sourceforge.net
[2] http://pyml.sourceforge.net
[3] http://secondstring.sourceforge.net

Data set	Train set	Threshold				
		0.1	0.3	0.5	0.7	0.9
Census	$\mathbf{W}_M$	0	100	96.2	73.4	67.9
	$\mathbf{W}_N$	0	0	100	100	100
Restau-	$\mathbf{W}_M$	100	98.5	4.5	0.19	0.2
rant	$\mathbf{W}_N$	0	0	100	100	100
DS-Gen	$\mathbf{W}_M$	0	100	100	100	100
1,000	$\mathbf{W}_N$	100	100	100	99.0	86.1
DS-Gen	$\mathbf{W}_M$	100	100	100	99.8	99.7
2,500	$\mathbf{W}_N$	100	100	100	99.4	92.0
DS-Gen	$\mathbf{W}_M$	100	100	100	98.0	96.5
5,000	$\mathbf{W}_N$	100	100	100	99.7	96.3
DS-Gen	$\mathbf{W}_M$	100	100	100	95.5	93.6
10,000	$\mathbf{W}_N$	99.2	99.7	100	99.9	98.3

Table 2: Quality of threshold-based training example selection. $\mathbf{W}_M$ denotes the match example training set, and $\mathbf{W}_N$ the non-match example training set. All result values are given as percentages.

Data set	Train set	Non-unique near.			Unique nearest		
		1%	5%	10%	1%	5%	10%
Census	$\mathbf{W}_M$	100	100	81.8	100	100	79.9
	$\mathbf{W}_N$	100	100	100	100	100	100
Restau-	$\mathbf{W}_M$	9.8	2.0	1.0	5.6	1.1	0.59
rant	$\mathbf{W}_N$	100	100	100	100	100	100
DS-Gen	$\mathbf{W}_M$	100	100	100	100	100	100
1,000	$\mathbf{W}_N$	100	96.7	95.5	100	95.9	95.5
DS-Gen	$\mathbf{W}_M$	100	100	100	100	100	100
2,500	$\mathbf{W}_N$	99.0	98.4	98.3	99.0	98.4	98.2
DS-Gen	$\mathbf{W}_M$	100	100	99.0	100	100	99.0
5,000	$\mathbf{W}_N$	100	99.8	99.5	99.7	99.7	99.6
DS-Gen	$\mathbf{W}_M$	100	99.0	75.4	100	98.6	74.1
10,000	$\mathbf{W}_N$	100	99.8	99.7	99.8	99.8	99.7

Table 3: Quality of balanced nearest-based training example selection.

Data set	Train set	Non-unique near.			Unique nearest		
		1%	5%	10%	1%	5%	10%
Census	$\mathbf{W}_M$	100	100	100	100	100	100
	$\mathbf{W}_N$	100	100	100	100	100	100
Restau-	$\mathbf{W}_M$	100	100	90.8	100	76.7	58.6
rant	$\mathbf{W}_N$	100	100	100	100	100	100
DS-Gen	$\mathbf{W}_M$	100	100	100	100	100	100
1,000	$\mathbf{W}_N$	100	96.7	95.5	100	95.9	95.5
DS-Gen	$\mathbf{W}_M$	100	100	100	100	100	100
2,500	$\mathbf{W}_N$	99.0	98.4	98.3	99.0	98.4	98.2
DS-Gen	$\mathbf{W}_M$	100	100	100	100	100	100
5,000	$\mathbf{W}_N$	100	99.8	99.5	99.7	99.7	99.6
DS-Gen	$\mathbf{W}_M$	100	100	100	100	100	100
10,000	$\mathbf{W}_N$	99.9	99.8	99.7	99.8	99.8	99.7

Table 4: Quality of imbalanced nearest-based training example selection.

corresponds to the *recall* measure as used in information retrieval. The reduction ratio measure, rr, is the number of record pairs generated by the blocking process divided by the number of all possible record pairs. For a linkage between two data sets, $\mathbf{A}$ and $\mathbf{B}$, $rr = 1.0 - |\mathbf{W}|/(|\mathbf{A}| \times |\mathbf{B}|)$ (with $\mathbf{W}$ the set of weight vectors generated in the comparison step), while for a deduplication $rr = 1.0 - 2|\mathbf{W}|/(|\mathbf{A}| \times (|\mathbf{A}| - 1))$. The more record pairs are removed by a blocking technique the higher the reduction ratio value becomes. However, reduction ratio does not take the quality of the generated candidate record pairs into account (how many are true matches or not). The ratio of matches to non-matches shown in Table 1 refers to the corresponding number of weights vectors in $\mathbf{W}$.

4.2 Quality Measures

The quality of the training examples selected in step one (shown in Tables 2, 3 and 4) are given as the percentage of correctly selected weight vectors, i.e. the percentage of true matches in the match examples training set $\mathbf{W}_M$, and the percentage of true non-matches in the non-match examples training set $\mathbf{W}_N$.

Due to the usually imbalanced distribution of matches and non-matches in the weight vector set $\mathbf{W}$, the commonly used accuracy measure is not suitable for assessing the quality of record linkage (Christen and Goiser 2007). The large number of non-matches would dominate the accuracy measure and yield results that are too optimistic. Instead, the F-measure, the harmonic mean of precision P and recall R, is used for measuring classifier quality: $F = 2PR/(P + R)$, with $P = TP/(TP + FP)$ and $R = TP/(TP + FN)$. TP is the number of true positives (true matched record pairs classified as matches), TN the number of true negatives (true non-matched record pairs classified as non-matches), FN the number of false negatives (true matched record pairs classified as non-matches), and FP the number of false positives (true non-matched record pairs classified as matches).

4.3 Results and Discussion

Tables 2, 3 and 4 show the quality of the training examples selected using the different approaches discussed in Section 3.1. As can be seen clearly, in many cases the selected weight vectors are of very high quality, i.e. they are all (or almost all) true matches and true non-matches. The threshold-based approach is problematic, as threshold values that are set too low will possibly result in no weight vectors being selected into a training set. The nearest-based approach overcomes this problem, especially the imbalanced nearest-based selection, which produced very good quality training data in almost all experiments. In the balanced nearest-based approach, it seems that too many weight vectors are selected into the match example training set $\mathbf{W}_M$, resulting in a loss of its quality, as with increasing training set size more false matches (false positives) will be selected due to the imbalanced numbers of true matches and non-matches in the weight vector set $\mathbf{W}$.

For the Census and Restaurant data sets, the 0% values for the threshold-based approach in Table 2 indicate that each of the record pairs compared had similar or equal values in at least one of the compared record attributes, while for the larger synthetic data sets there were record pairs that had no similar attribute values at all. This means that the blocking step for the synthetic data sets could be improved, as record pairs that have no similar attribute values at all clearly should not be compared using the computationally expensive comparison functions.

As can be seen from Table 5 and Figures 3, 4 and 5, using the automatically selected training example sets for classification produced results of a wide variety, ranging from almost as good as the supervised optimal threshold and SVM classifiers, to F-measure values much lower than those of k-means clustering. With most data sets, the linear SVM classifier achieved the best F-measure results, outperforming the optimal threshold classifier. Of the two-step approaches, the threshold based approach seems to be very sensitive to the chosen threshold value, while with the nearest-based approach, imbalanced training set size outperforms balanced training set size in most cases, often achieving significantly better results than k-means clustering. For the balanced nearest-based approach, there seems to be a general trend that smaller training set size results in better classi-

Classification approach	Data sets					
	Census	Restaurant	DS-Gen 1,000	DS-Gen 2,500	DS-Gen 5,000	DS-Gen 10,000
Optimal threshold	0.784	0.839	0.900	0.846	0.813	0.787
SVM	0.785	0.466	0.944	0.917	0.884	0.829
K-means clustering	0.434	0.002	0.802	0.814	0.763	0.213
Threshold-0.1	0.000	0.000	0.000	0.844	0.808	0.750
Threshold-0.3	0.000	0.000	0.857	0.809	0.735	0.655
Threshold-0.5	0.187	0.001	0.711	0.609	0.527	0.751
Threshold-0.7	0.171	0.704	0.826	0.816	0.744	0.492
Threshold-0.9	0.149	0.379	0.779	0.681	0.688	0.458
Nearest-1%, NU, B	0.566	0.001	0.854	0.821	0.728	0.500
Nearest-5%, NU, B	0.643	0.001	0.865	0.815	0.573	0.199
Nearest-10%, NU, B	0.317	0.001	0.840	0.757	0.344	0.080
Nearest-1%, U, B	0.566	0.002	0.863	0.834	0.741	0.515
Nearest-5%, U, B	0.500	0.002	0.861	0.813	0.582	0.203
Nearest-10%, U, B	0.271	0.002	0.841	0.757	0.348	0.087
Nearest-1%, NU, IB	0.567	0.044	0.865	0.815	0.780	0.738
Nearest-5%, NU, IB	0.644	0.012	0.851	0.823	0.805	0.751
Nearest-10%, NU, IB	0.410	0.002	0.830	0.817	0.797	0.739
Nearest-1%, U, IB	0.568	0.008	0.858	0.806	0.781	0.757
Nearest-5%, U, IB	0.511	0.005	0.849	0.822	0.807	0.756
Nearest-10%, U, IB	0.388	0.003	0.832	0.815	0.799	0.747

Table 5: F-measure classification results. 'U' and 'NU' denote the unique and non-unique weight vector selection, respectively, and 'B' and 'IB' the balanced and imbalanced training set size selection. Note that 'Optimal threshold' and 'SVM' are supervised classification techniques, while all other approaches are unsupervised.

fication quality compared to larger training sets. No such trend is visible for the imbalanced nearest-based approach. There is also no clear advantage or disadvantage for using unique or non-unique nearest-based selection of training examples.

These initial results indicate that the proposed two-step approach to automatic record pair classification is feasible and can achieve linkage quality almost as good a fully supervised classification. Specifically, the nearest-based selection of match and non-match training example sets can automatically generate training data of high quality. However, more investigation is needed for the second step of the proposed approach; on how to best use the generated training example sets for classification. More experiments using different data sets and additional classifiers have to be conducted in order to validate the general applicability of the proposed approach to a wide range of data with different characteristics.

5 Conclusions and Future Work

In this paper, a novel two-step approach to record pair classification has been presented that aims to automate the record linkage process. This approach combines an automatic selection of training examples, that with high likelihood are either true matches or true non-matches, with a traditional supervised classifier. Initial experiments on a range of data sets showed promising results, in certain cases achieving F-measure values almost as good as a fully supervised linear SVM classifier, but generally better than an unsupervised k-means clustering approach.

There are two main extensions to the basic approach presented here that will be investigated in the near future. First, rather than only using a classifier once in the second step to classify all weight vectors in $\mathbf{W}_T$, an iterative approach, similar to PEBL (Yu et al. 2002), will be explored. The basic idea is that in each iteration the strongest classified matches and non-matches in $\mathbf{W}_T$, i.e. the weight vectors furthest away from the decision boundary, will be added to the training sets $\mathbf{W}_M$ and $\mathbf{W}_N$. This process is repeated until a certain stopping is fulfilled.

Second, similar to the inclusion of 'spy' documents in the S-EM approach to partially supervised text classification (Liu et al. 2002), adding randomly sampled weight vectors from the 'gap' between the selected matches and non-matches into the training example sets should improve the overall classification accuracy. Random sampling should be conducted such that weight vectors closer to the exact match vector $\mathbf{m}$ are more likely included into the set of match training examples, $\mathbf{W}_M$, while weight vectors that contain mainly dissimilarity values should more likely be included into the set of non-match training examples, $\mathbf{W}_N$. This idea is currently being implemented and results will be reported elsewhere.

Additionally, the effects of using different approximate string comparison techniques (Christen 2006) on the proposed approach will also be investigated.

Other future work will include a scalability and complexity analysis, as well as timing measurements, of the proposed approach. Given that only a fraction of all weight vectors are selected into the two training sets in step one, the time required to train a classifier in the second step of the proposed approach should be small compared to using all weight vectors for supervised classification or clustering.

6 Acknowledgements

This work is supported by an Australian Research Council (ARC) Linkage Grant LP0453463 and partially funded by the New South Wales Department of Health. The author would like to thank Paul Thomas for proof-reading this paper.

References

Basu, S., Banerjee, A. & Mooney, R.J. (2002), Semi-supervised clustering by seeding, *in* 'International Conference on Machine Learning' (ICML'02), Sydney, Australia, pp. 19–26.

Baxter, R., Christen, P. & Churches, T. (2003), A comparison of fast blocking methods for record linkage, *in* 'ACM SIGKDD Workshop on Data Cleaning, Record Linkage and Object Consolidation', Washington DC, pp. 25–27.

Bhattacharya, I. & Getoor, L. (2007), 'Collective entity resolution in relational data', *ACM Transac-*

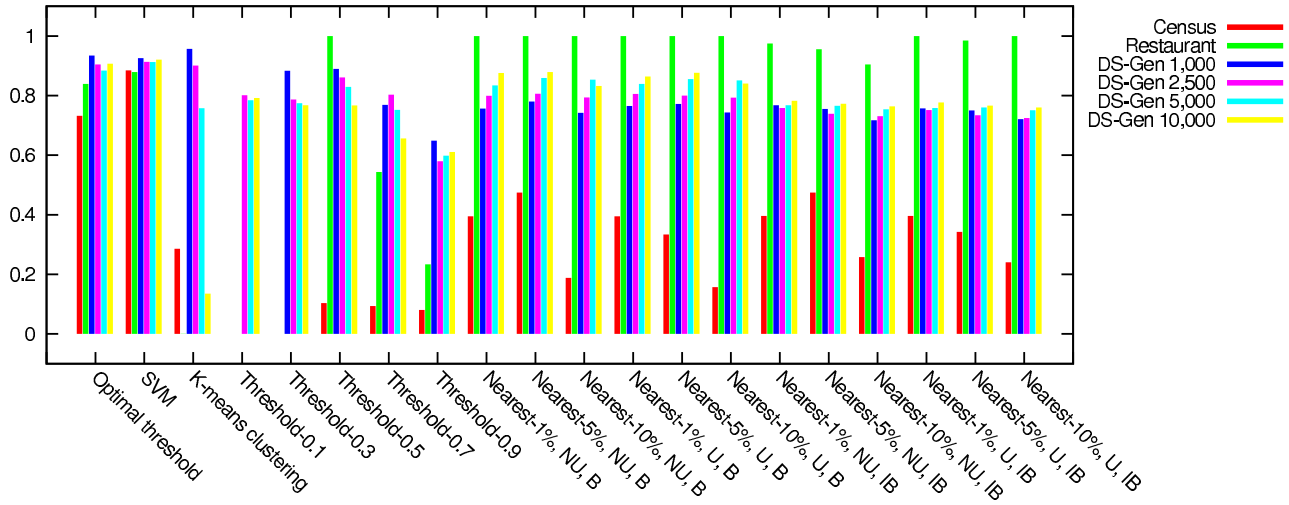

Figure 3: Precision results for all data sets and classification methods.

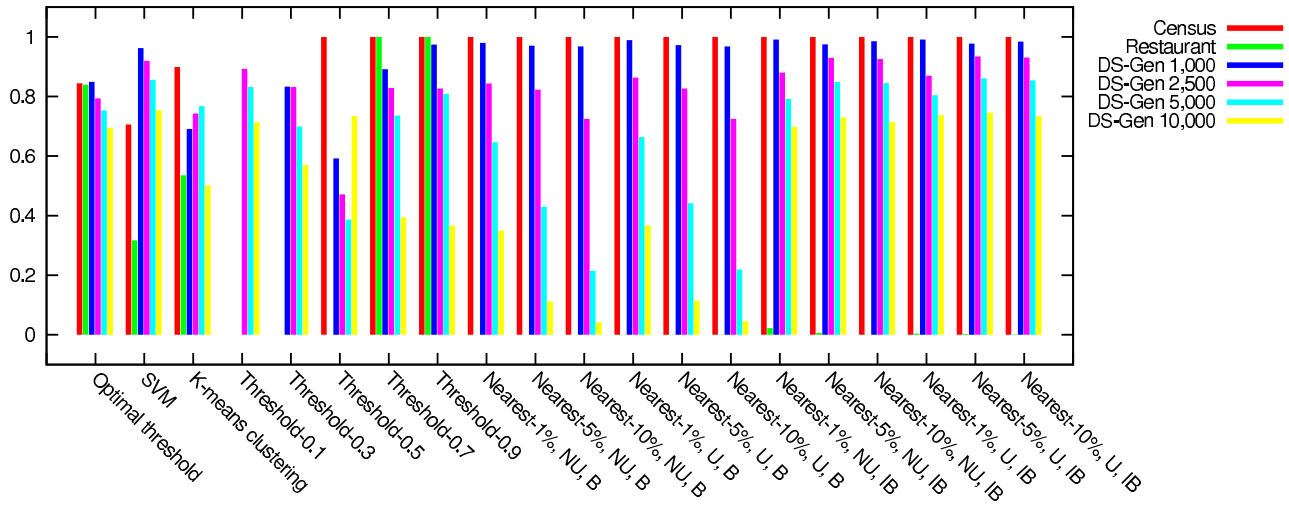

Figure 4: Recall results for all data sets and classification methods.

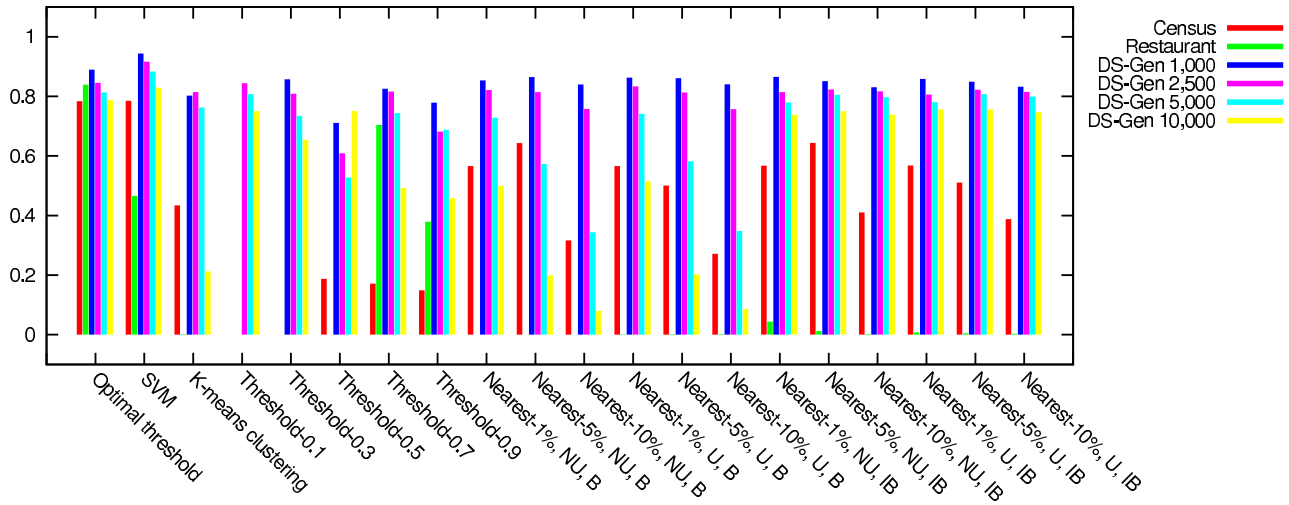

Figure 5: F-measure results for all data sets and classification methods.

tions on Knowledge Discovery from Data (TKDD), vol. 1, no. 1.

Bilenko, M. & Mooney, R.J. (2003), Adaptive duplicate detection using learnable string similarity measures, *in* 'ACM International Conference on Knowledge Discovery and Data Mining' (SIGKDD'03), Washington DC, pp. 39–48.

Bilenko, M., Basu, S. & Sahami, M. (2005), Adaptive product normalization: Using online learning for record linkage in comparison shopping, *in*

'IEEE International Conference on Data Mining' (ICDM'05), Houston, Texas, pp. 58–65.

Chang, C.-C. & Lin, C.-J. (2001), LIBSVM: a library for support vector machines, manual. Department of Computer Science, National Taiwan University. Software available at: `http://www.csie.ntu.edu.tw/~cjlin/libsvm`.

Christen, P., Churches, T. & Hegland, M. (2004), Febrl – A parallel open source data linkage system, *in* 'Pacific-Asia Conference on Knowledge Discovery and Data Mining' (PAKDD'04), Sydney, Springer LNAI 3056, pp. 638–647.

Christen, P. (2005), Probabilistic data generation for deduplication and data linkage, *in* 'International Conference on Intelligent Data Engineering and Automated Learning' (IDEAL'05), Brisbane, Springer LNCS 3578, pp. 109–116.

Christen, P. (2006), A comparison of personal name matching: techniques and practical issues, *in* 'Workshop on Mining Complex Data' (MCD), held at IEEE ICDM'06, Hong Kong.

Christen, P. & Goiser, K. (2007), Quality and complexity measures for data linkage and deduplication, *in* F. Guillet & H. Hamilton, eds, 'Quality Measures in Data Mining', Springer Studies in Computational Intelligence, vol. 43, pp. 127–151.

Churches, T., Christen, P., Lim, K. & Zhu, J.X. (2002), 'Preparation of name and address data for record linkage using hidden Markov models', *BioMed Central Medical Informatics and Decision Making*, vol. 2, no. 9.

Clarke, D.E. (2004), 'Practical introduction to record linkage for injury research', *Injury Prevention*, vol. 10, pp. 186–191.

Cohen, W.W. & Richman, J. (2002), Learning to match and cluster large high-dimensional data sets for data integration, *in* 'ACM International Conference on Knowledge Discovery and Data Mining' (SIGKDD'02), Edmonton, pp. 475–480.

Cohen W.W., Ravikumar P. & Fienberg S.E. (2003), A comparison of string distance metrics for name-matching task, *in* 'IJCAI-03 workshop on information integration on the Web' (IIWeb-03), Acapulco, pp. 73–78.

Elfeky, M.G., Verykios, V.S. & Elmagarmid, A.K. (2002), TAILOR: A record linkage toolbox, *in* 'International Conference on Data Engineering' (ICDE'02), San Jose, pp. 17–28.

Fellegi, I.P. & Sunter, A.B. (1969), 'A theory for record linkage', *Journal of the American Statistical Society*, vol. 64, no. 328, pp. 1183–1210.

Gill, L. (2001), 'Methods for automatic record matching and linking and their use in national statistics', *National Statistics Methodology Series*, no. 25, National Statistics, London.

Goiser K. & Christen, P. (2006), Towards automated record linkage, *in* 'Australasian Data Mining Conference' (AusDM'06), Sydney, Conferences in Research and Practice in Information Technology (CRPIT), vol. 61, pp. 23–31.

Gu, L. & Baxter, R. (2006), Decision models for record linkage, *in* 'Selected Papers from AusDM', Springer LNCS 3755, pp. 146–160.

Kelman, C.W., Bass, J. & Holman, D. (2002), 'Research use of linked health data – A best practice protocol', *Aust NZ Journal of Public Health*, vol. 26, pp. 251–255.

Liu, B., Lee, W.S., Yu, P.S. & Li, X. (2002), Partially supervised classification of text documents, *in* 'International Conference on Machine Learning' (ICML'02), Sydney, Australia, pp. 387–394.

Liu, B., Dai, Y., Li, X., Lee, W.S. & Yu, P.S. (2003), Building text classifiers using positive and unlabeled examples, *in* 'IEEE International Conference on Data Mining' (ICDM'03), Melbourne, Florida, pp. 179–186.

McCallum, A., Nigam, K. & Ungar, L.H. (2000), Efficient clustering of high-dimensional data sets with application to reference matching, *in* 'ACM International Conference on Knowledge Discovery and Data Mining' (SIGKDD'00), Boston, pp. 169–178.

Nahm, U.Y., Bilenko, M. & Mooney, R.J. (2002), Two approaches to handling noisy variation in text mining, *in* 'ICML'02 workshop on text learning' (TextML'02), Sydney, Australia, pp. 18–27.

Neiling, M. (2005), Identification of real-world objects in multiple databases, *in* '29th Annual Conference of the Gesellschaft für Klassifikation e.V.', University of Magdeburg, pp. 63–74.

Nigam, K., McCallum, A.K., Thrun, S. & Mitchell, T. (2000), 'Text classification from labeled and unlabeled documents using EM', *Machine Learning*, vol. 39, no. 2, pp. 103–134.

Rahm, E. & Do, H.H. (2000), 'Data cleaning: Problems and current approaches', *IEEE Data Engineering Bulletin*, vol. 23, no. 4, pp. 3–13.

Sarawagi S., & Bhamidipaty A. (2002), Interactive deduplication using active learning, *in* 'ACM International Conference on Knowledge Discovery and Data Mining' (SIGKDD'02), Edmonton, Canada, pp. 269–278.

Tejada S., Knoblock C.A. & Minton S. (2000), Learning domain-independent string transformation weights for high accuracy object identification, *in* 'ACM International Conference on Knowledge Discovery and Data Mining' (SIGKDD'02), Edmonton, Canada, pp. 350–359.

Wang, G., Chen, H., Xu, J.J. & Atabakhsh, H. (2006), 'Automatically detecting criminal identity deception: An adaptive detection algorithm', *IEEE Transactions on Systems, Man and Cybernetics (Part A)*, vol. 36, no. 5, pp. 988–999.

Winkler, W.E. (2000), 'Using the EM algorithm for weight computation in the Fellegi-Sunter model of record linkage', *Technical report RR2000/05*, US Bureau of the Census.

Winkler, W.E. (2004), 'Methods for evaluating and creating data quality', *Information Systems*, Elsevier, vol. 29, no. 7, pp. 531–550.

Winkler, W.E. (2006), 'Overview of record linkage and current research directions', *Technical report RR2006/02*, US Bureau of the Census.

Yu, H., Han. J. & Chang, K.C.C. (2002), PEBL: Positive example based learning for Web page classification using SVM, *in* 'ACM International Conference on Knowledge Discovery and Data Mining' (SIGKDD'02), Edmonton, Canada, pp. 239–248.