Improving Record Linkage Through Pedigrees

Burdette N. Pixton
Brigham Young University - Provo

IMPROVING RECORD LINKAGE THROUGH PEDIGREES

by

Burdette Pixton

A thesis submitted to the faculty of

Brigham Young University

in partial fulfillment of the requirements for the degree of

Master of Science

Department of Physical

and Mathematical Sciences

Brigham Young University

August 2006

BRIGHAM YOUNG UNIVERSITY

GRADUATE COMMITTEE APPROVAL

of a thesis submitted by

Burdette Pixton

This thesis has been read by each member of the following graduate committee and by majority vote has been found to be satisfactory.

Date	Christophe Giraud-Carrier, Chair
Date	Dan Ventura
Date	Quinn Snell

ABSTRACT

IMPROVING RECORD LINKAGE THROUGH PEDIGREES

Burdette Pixton

Department of Physical

and Mathematical Sciences

Master of Science

Record linkage, in a genealogical context, is the process of identifying individuals from multiple sources which refer to the same real-world entity. Current solutions focus on the individuals in question and on complex rules developed by human experts. Genealogical databases are highly-structured with relationships existing between the individuals and other instances. These relationships can be utilized and human involvement greatly minimized by using a filtered structured neural network. These neural networks, using traditional back-propagation methods, are biased in a way to make the network human readable. The results show an increase in precision and recall when pedigree data is available and used.

ACKNOWLEDGMENTS

I would like to acknowledge Dr. Randy Wilson from FCHD for his ideas in utilizing pedigree data and in securing the data used in this Thesis. I would also like to thank Dr. Christophe Giraud-Carrier whose tireless effort, support and thoughts taught me how to do research and made this work possible. Lastly, I thank my beautiful wife, Michelle, for her love and support in taking care of me throughout this process.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

Record linkage is the process of integrating information from two or more independent sources. It ensures that records believed to represent the same object are matched and treated as a single entity. Record linkage is a subset of a larger problem known as the object-identity problem. The object-identity problem is common in large or distributed databases. The duplication of objects within a database or across several databases causes a host of problems, including increased storage requirements, performance degradation, undesired bias towards duplicates in statistical analyses [1], and non-deterministic behavior if duplicates do not agree on all data fields. Under such conditions, it is not only necessary to detect duplicates but also to merge the corresponding records.

In addition to solving the above technical problems, identifying and merging duplicates has beneficial side-effects. For example, in the medical domain, one may be able to discover hereditary conditions [2] or to highlight links among various pharmaceutical drugs and medical conditions by detecting identical patients across a pharmacy's and a physician's databases. In the domain of genealogy, where record linkage seeks specifically to identify whether or not individuals belonging to different pedigrees refer to the same person [2], one may be able to obtain missing information from the work of others. Consider two genealogical researchers R_1 and R_2 who are each examining a pedigree represented in Figures 1.1 and 1.2.

Assume that records K and E are found to refer to the same individual. The two pedigrees can now be merged and both researchers immediately gain access to new information, from missing details about individuals to completely new individuals.

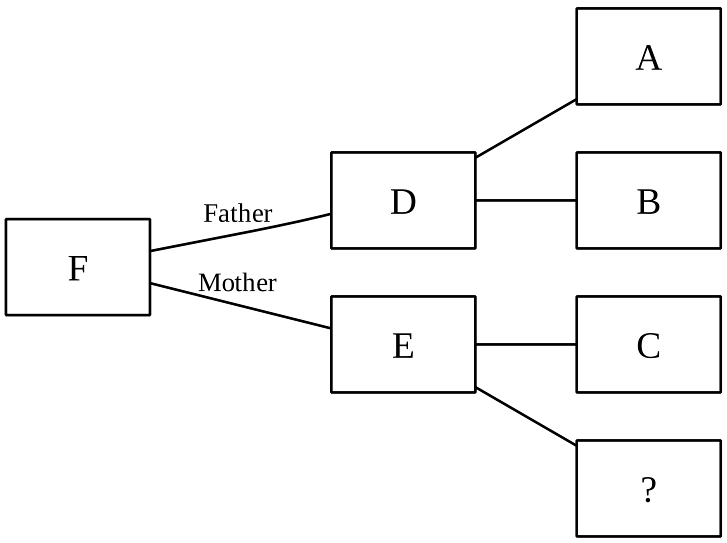

Figure 1.1: R_1's Pedigree

Here, for example, R_2 discovers F, a child of K, from information in R_1's pedigree. R_1 is similarly benefited by discovering J, the mother of E, from information in R_2's pedigree.

Genealogical researchers greatly benefit from merging pedigrees with other researchers as it not only provides additional knowledge about certain instances, but can also be used to detect erroneous information and eliminate duplication of work. Furthermore, there is a kind of snowball effect to record linkage. As more pedigrees are merged, more knowledge becomes available about existing pedigrees. This knowledge in turn can be used to merge more pedigrees, and so on.

As the custodian of the largest source of genealogical data, the Church of Jesus Christ of Latter-day Saints' Family and Church History Department (FCHD) has a keen interest in accurate record linkage. The FCHD's main objectives are to (a) guarantee the integrity and overall quality of the available data, (b) facilitate researchers' work by merging together complementary information, and (c) avoid duplication of ordinance work.

Traditionally, record linkage strategies have focused on designing similarity

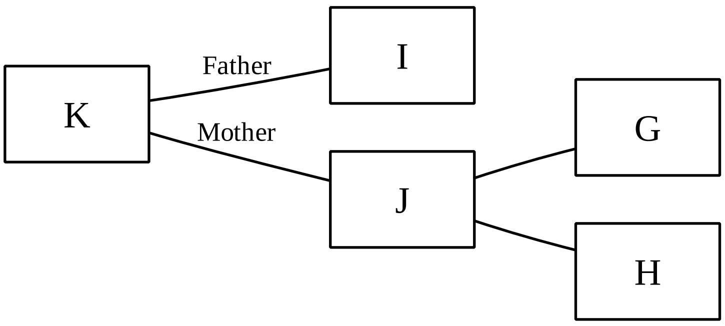

Figure 1.2: R_2's Pedigree

metrics and combining them intelligently to produce an overall score whose value is used to predict whether two individuals are the same. The focus is generally on the attributes of individuals (e.g., name, date of birth), which implicitly presupposes that such information is available. Although this is often the case, there are also many situations, arising naturally from the genealogical research process, where details about individuals are lacking. For example, suppose a genealogist discovers a new person, $newPerson$, in a given pedigree. At this stage, not much research has been done on $newPerson$ by the genealogist; only its existence and placement in the pedigree has occurred. A natural next step for the genealogist would be to determine whether any other researcher has discovered (and documented) $newPerson$ before. By locating this information, the genealogist can save time by merging both findings.

Current record linkage strategies do not facilitate this kind of discovery. They require the genealogist to first research the details of $newPerson$ and only then attempt to find a potential match in other pedigrees. Our proposed approach, Mining and Linking for Successful Information eXchange (MAL4:6), is an attempt at alleviating this situation by explicitly taking into account the pedigrees when matching individuals. MAL4:6 traverses both pedigrees in parallel to judge the similarities of each pair of corresponding individuals. These measurements are combined and a final classification is predicted.

Chapter 2

Related Work

Algorithms that perform record linkage generally involve the four steps shown in Figure 2.1. Step 1 consists of comparing the individual attributes of the instances of interest. The attributes of a genealogical database would include things such as names, locations and dates. These attributes are generally stored as strings, and thus string metrics are often used for their comparisons. Step 2 involves quantifying the previous comparisons. The numerical comparison provided by the string metric is mapped to a score through a set of rules. In step 3, all of the scores from the various attributes are used and combined to produce an overall similarity. This process attempts to aggregate the knowledge from the various pieces to determine an overall similarity for the entities. Finally, in step 4, a classification is made for the entities. In genealogical terms, this would mean predicting that the two entities involved refer to the same individual or not.

Step 1. Compare the attributes
Step 2. Quantify the comparisons
Step 3. Combine the scores from the comparisons
Step 4. Compare against a threshold

Figure 2.1: Steps in Record Linkage

2.1 Basic Definitions

To assist with the rest of this discussion and throughout the thesis, we provide the following set of basic definitions.

Let $\Gamma = \{\gamma_1, \gamma_2, ..., \gamma_n\}$ be the set of attributes that characterize individuals. In practice, each γ_k represents some piece of information about individuals, such as first name, last name, date of birth, place of birth, etc.

An individual x is denoted by an ordered set of attribute value pairs $< \gamma_k{:}v_k >$ (e.g., $< firstname$: John, $lastname$: Smith, $\ldots >$). We use γ_k^x to refer to the value of attribute γ_k of an individual x. Two individuals, x and y, that are being compared against each other consititute a pair p.

For each attribute, γ_k, there is an associated similarity measure, sim_{γ_k}, which is selected based upon the type of γ_k (see Table 3.1 in Chapter 3). Then, for any pair of individuals, $p =< x, y >$, we define a similarity function as follows.

$$sim_p(x, y) = \sum_{k=1}^{|\Gamma|} \alpha_k sim_{\gamma_k}(\gamma_k^x, \gamma_k^y) \tag{2.1}$$

where the α_k's are optional weights applied to the attributes.

In some cases, similarity measures are used in a mapping process (described as step 3 in Figure 2.1) to create a new score for the attribute. In these circumstances, e_k will be used to denote a score for attribute γ_k.

2.2 Record Linkage Standards

Entities in standard databases do not generally share a common relationship. Each row is independent from other entries. This limits the algorithms used to detect matches to the data contained in the row. A naive approach is to compare each data element (cell) of a row with every other row in the database. The rows whose data matches exactly along predetermined cells are removed as duplicates. This method is a straightforward fix and removes the obvious duplicates. It is, however, very weak in that it may not find all of the duplicates. Records, whose data differ, even slightly, will not be located with this method.

Detecting misspellings and standardizing the data to a common form solves some of the problems with the naive approach. String metrics, discussed in the next chapter, are used to assign a numerical value by judging the similarity between two strings. Not all differences among like data rows are caused by errors. Some of the differences are intentional. Individual data elements may refer to the same real-world entity but be in a different form. Proper names often have different forms within the same database. The names Jen and Jenny, for example, refer to the same name but are spelled differently. They should be standardized to the same root name of Jennifer. Locations often need to be standardized too. For example, the two locations of Murray, UT and Holiday, UT are realistically different places but should probably be considered part of the larger Salt Lake City in a national database. Geographical and name dictionaries are routinely used in determining root or base forms of these types of nouns.

The current record linkage system used by the FCHD uses standardization and relies on a scoring mechanism based on sophisticated, hand-crafted comparison rules (e.g., if the birth year matches within some tolerance, a positive score is awarded; if it differs, a negative score). This approach has some basis in the Probabilistic Record Linkage formula, which was introduced by Newcomb [3] and formalized by Fellegi & Sunter [4].

2.2.1 Fellegi-Sunter

Many record linkage techniques have a basis in the principles introduced by Newcomb [3] in 1959. Ten years later, in 1969, Fellegi & Sunter [4] gave a formal mathematical account of Newcomb's work. Winkler, in [5], highlights three key reasons why the Fellegi-Sunter approach is useful: (1) it estimates outcome probabilities, (2) it provides estimates on error rates without manual intervention, and (3) it estimates threshold values based on those error rates. The method relies on the use of probabilities on attributes which entities may have in common. A detailed explanation of how this is implemented is in [4] and [6].

2.2.2 Bayesian Systems

Algorithms based on Fellegi & Sunter's method to predict threshold values are often extended to include weights on the various attributes. Weights allow greater (or lesser) importance to be apportioned to certain data elements. Many techniques may be used to calculate the values for the weights, including machine learning algorithms, rule-based approaches and statistical analysis. A Bayesian network is commonly used in the statistical approach.

Bayesian networks are built using probabilities computed from a training set. Weights, in this case, are defined by Equation 2.2.

$$\begin{aligned} w_i &= \ln[P(M|e_k)] \\ &= \ln\left[P(M)\right] + \ln\left[\frac{P(e_k|M)}{P(e_k)}\right] \end{aligned} \tag{2.2}$$

$P(M)$ is a constant and is the probability of a match occurring between two instances given a pair p; it is calculated from the distribution of the test set. $P(e_k)$ is the probability that given a record, attribute k will have a score of at least e; it is estimated from a training set. Ideally, the probability, $P(e_k|M)$, that an attribute will have a score of at least e given that it is a match is calculated from known samples.

This weight of Equation 2.2 is applied to an unknown pair p based on the similarity of the attributes that exist between p's individuals, as shown in Equation 2.3.

$$e_k = \begin{cases} w_i & \text{if } sim_{\gamma_k}(p.\gamma_k : p_k) > \delta, \\ -w_i & \text{otherwise.} \end{cases} \tag{2.3}$$

It is important to note that the similarity score produced by $sim_{\gamma_k}(p.\gamma_k : p_k)$ is not applied to e_k. It is only used to determine a value for e_k. Finally, the δ value, although it can be learned, is generally defined by the user.

2.2.3 Handcrafted Fellegi-Sunter

Many current approaches perform the steps mentioned in Figure 2.1 by relying on human knowledge and intuition. Step 1 is often performed using an arbitrary string metric to compare the attributes that an expert deems to be important. Scores (w_i) are determined using a mapping function like the one described in Equation 2.3. The value for w_i can be manipulated in order to increase precision/recall. The various values that w_i can assume are often determined by guess-work and humans learn ideal weights through trial and error. This methodology performs well on some sets because humans tend to have intuition about which attributes are important in determining if pairs are a match. For instance, a human can quite easily decide that a low w_i is appropriate if two genders match. However, if two individuals do not share the same gender a very large negative weight should be applied.

The Bayesian and handcrafted method of the Fellegi-Sunter equations often ignore the similarity value that is generated between each of the attributes for two individuals. This value is ignored in the sense that it is only used to compare against a threshold δ. The mapped score, not the similarity measurement, is used in calculating a final similarity score for the 2 individuals.

The final similarity score is computed using a set of rules which follow the format described in Equation 2.3. These rules consist of many criteria and thresholds (δ). Each of these rules returns a score which is aggregated with other scores to determine a prediction. The final two steps (aggregating and comparing the scores) are typically accomplished similarly to how step 2 is done. A set of rules or a function can combine the various scores. The threshold (δ) in step 4 can be adjusted similarly to w_i.

A rule-based approach relies heavily on human intervention. The scores returned and the criteria are predetermined. Rules are maintained manually and additional rules may be added to the set to increase performance. A set of rules may therefore be overfitted by human intervention to perform well on a certain test set. It should be noted that overfitting may take place with all models, whether built by humans or machines, especially when limited data sets are available for experimentation.

Probabilistic methods depend on the existence and correctness of an individual's data. Missing or erroneous attributes can greatly impact the performance of such methods. Probabilistic methods attempt to mitigate this by adjusting the weights of attributes which are often missing or wrong in a training set. Prior knowledge can be very useful here. However, by bringing the weights closer to zero (thus limiting their importance), additional emphasis is placed on the remaining attributes.

2.3 Problems with Current Approaches

Current approaches do not fully utilize the structure that is associated with genealogical records. A genealogical database is rich in structure with individuals often having links or connections with other individuals in the database through family relationships. Harvesting the attributes of other instances can help spread the dependency across more values and thus lessen the impact of erroneous and missing data. In genealogical databases, the process of creating additional data is relatively straightforward. A relative's data can be used to generate additional attributes. For example, the fact that two individuals have the same parents listed will have an impact in determining whether or not the persons are the same.

The human element involved in the handcrafted process makes it difficult to add an arbitrary number of new parameters. The parameters that the system uses, rules and weights, need to be maintained for each attribute. Adding new attributes to the system requires not only new weights and rules but also adjustments to the existing parameters. Adjustments to all parameters are made depending on the presence of other attributes. This set of rules may be easy to maintain if the number of attributes is small or if they commonly exist. This approach, however, does not work very well if attributes tend to be missing. A set of rules would be needed for different distributions of data. For instance, a set of rules R_1 may be appropriate if an individual has much ancestry data, but a different set R_2 may work better with descendents in the pedigree. Depending on the granularity of the desired attributes, the set of rules can be very large and difficult to maintain.

MAL4:6 eliminates the need for human interaction by automating all of the steps outlined in Figure 2.1. Automatically generating a set of mapping rules would be difficult for MAL4:6. Mapping rules are strongly dependent upon the aggregation method used in the third step. The result of this is that for each algorithm tested in the third step, a different set of mapping rules would potentially need to be learned. Judging the success of different algorithms would then be difficult as success or failure could be largely attributed to the mapping stage. For example, a given algorithm may have performed better if its associated mapping function had separated the data more uniformly.

We simplify the problem of creating rule sets or mapping functions for step 2 by using the string metric score of step 1 directly. In other words, $e_k = sim(\gamma_k)$. This, of course, places greater emphasis on the metric chosen. We return to this in Chapter 3. Finally, the third step of the process is also automated. Machine learning algorithms are used for combining the various similarity scores and determining if a pair is a match. One of the side benefits of this automation is that new models can be adapted quickly when parameter adjustments (e.g., introduction of new attributes) are needed.

Chapter 3

Similarity Function

Given a pair of entities, the first step in the record linkage process (see Figure 2.1) is to compare their data elements against each other. Genealogical records consist of three different types of data, namely: names, dates and locations. Because these data types are often stored as strings, a way is needed to compute the similarity between two given strings. Simply deciding if two strings are equal (or not) is insufficient. It is necessary to know how close they are to each other. String metrics provide a method to judge the similarity on a continuous scale.

Many string metrics have been developed over time, and they each base their similarity score on a number of different properties, such as spelling, phonetics and compression. Edit distance metrics, such as Levenshtein's distance, developed in 1965, focus on string transformations [7] and compare strings on a letter by letter basis. Similarity metrics are akin to edit distance metrics but they focus on groups of letters instead of individual letter placement. Jaro [8, 9], and a variation known as Jaro-Winkler [10], are the two prime examples of this type of metric.

Some strings, such as names and locations, are commonly misspelled due to ignorance, rather than typos. For example, John and Jon sound the same but are spelled differently. These strings, when compared with Levenshtein or Jaro, would receive a low score. These type of 'misspellings' are common in genealogical data, however, as recorders are sometimes left up to how they would spell a certain name. Phonetic metrics focus on word sound rather than word spelling. With such metrics, Jon and John would receive a high score. Soundex [11] is the primary example of this type of metric.

The choice of string metric does not typically play a critical role in current record linkage approaches, since the actual value, $sim(\gamma_k)$, is only used as input to a mapping process (a set of functions) that produce the second, actionable score, e_k. In order to reduce the human involvement in the process and allow MAL4:6 the opportunity to manipulate the attribute weights on its own, no mapping is performed; only $sim(\gamma_k)$ is used directly. This decision has two major impacts on the system. It eliminates the need for MAL4:6 (or a human expert) to learn/build a set of mapping rules, and it places greater emphasis on the metric chosen.

3.1 String Metrics in MAL4:6

MAL4:6 must choose a metric that can adequately distinguish two strings. Obviously, a metric would be considered good if it gives high scores to strings which are very similar and low scores otherwise. Determining which metric works best is not a trivial task. Furthermore, the different types of string metrics may perform differently depending on the data type encountered. For instance, using an edit distance metric may perform well on dates, but a phonetic metric may do better on names.

A set of experiments were conducted to determine an appropriate metric for each attribute. Metrics from the edit distance, similarity and phonetic classes were selected and used in the experiments. Characteristics or trends involving these data types (i.e., names, dates and locations) may lend themselves to the strengths of a particular metric. In order for MAL4:6 to be successful, it is critical that good metrics are chosen for the various data elements contained in the records.

3.1.1 A Universal Metric

Assume h_i,h_j,u_a, and u_b are strings from the set of matches (H) and mismatches (U). A metric is considered to be a universal classifier if it satisfies Equation 3.1.

$$\forall h_i, h_j \forall u_a, u_b \; sim(h_i, h_j) > sim(u_a, u_b) \tag{3.1}$$

In other words, strings which originate from matches should have a higher similarity score than strings originating from mismatches.

A universal classifier would not need any learning or training. However, due to misspellings and erroneous information in H, coincidences in U, and errors in the data sets, such a metric does not and may never exist.

3.1.2 Creating a Designer Metric

A series of experiments was performed to test how the different string metrics performed on various types of genealogical data provided by the FCHD. We consider Soundex [11], Jaro's metric [8, 9], Jaro-Winkler's metric [10], Dice's approach [12, 13], binary discrimination (i.e., same vs. different) and the inverse of the 1-norm (i.e., 1 over the absolute value of the difference; 1 if values are the same). The inverse of the 1-norm is used primarily with numerical data. For this reason, the inverse of the 1-norm was only applied to day numbers and years. Other string metrics were also applied to these attributes, however, to compare their performance on numerical data.

MAL4:6 calculates a similarity score for each attribute with each metric across all pedigrees in H and U. This illustrates a basic assumption which is made throughout the thesis. The corresponding relatives of a pair belong to the same classification (*match* or *mismatch*) as p. For instance, if x and y are matches and therefore in set H, then the pair made up of their respective fathers is also a match. While this is a law of nature, errors contained in the associated pedigrees may cause this assumption not to hold.

MAL4:6 stores the similarity score according to its classification (match, non-match), the attribute, and the metric used. Upon completion of each pair in the sets, an average score is computed. The optimal metric for each attribute is computed using the algorithm shown in Figure 3.1.

The outcome from these experiments is given in Table 3.1, where the best metric is listed for each type of attribute. These results give rise naturally to a heterogeneous metric for MAL4:6.

```
Input: sets $H$ and $U$
For each attribute $a$
    $max_{am} = 0$
    For each metric $m$
        $d_{am} = \text{Avg}_H(m(a))$ - $\text{Avg}_U(m(a))$
        If $d_{am} > max_{am}$
            $max_{am} = d_{am}$
            $optimalMetric_a = m$
```

Figure 3.1: Optimal Metric Algorithm

3.1.3 Validation of Heterogeneous Metric

In order to validate our selection of the heterogeneous metric, it is necessary to compare its accuracy with that of the homogeneous approaches. This initial implementation of MAL4:6 is referred to as version 0.1. As a first attempt, each metric is used across the attributes and an overall score (step 3 in the process) is computed as a simple average as follows.

$$sim(x, y) = \frac{\sum_{k=1}^{|\Gamma|} \alpha_k sim_{\gamma_k}(x.\gamma_k : v_k, y.\gamma_k : v_k)}{N}$$

where N is the number of attributes with values in both x and y. Note that $N \leq |\Gamma|$. In the homogeneous case, sim_{γ_k} is the same for all attributes and is the metric being tested. In the heterogeneous case, sim_{γ_k} is the optimal metric for the type of attribute k, as given in Table 3.1. Further, we define $sim_{\gamma_k}(x.\gamma_k : x_k, y.\gamma_k : y_k) = 0$ whenever either of x_k or y_k is missing.

Empirical results suggest that our composite metric is slightly superior to all of the homogeneous metrics. Figure 3.2 graphs the ROC curves parameterized by the similarity threshold for matching pairs. To highlight the differences, Figure 3.3 displays the corresponding *log* curves for the three best performing metrics in the study.

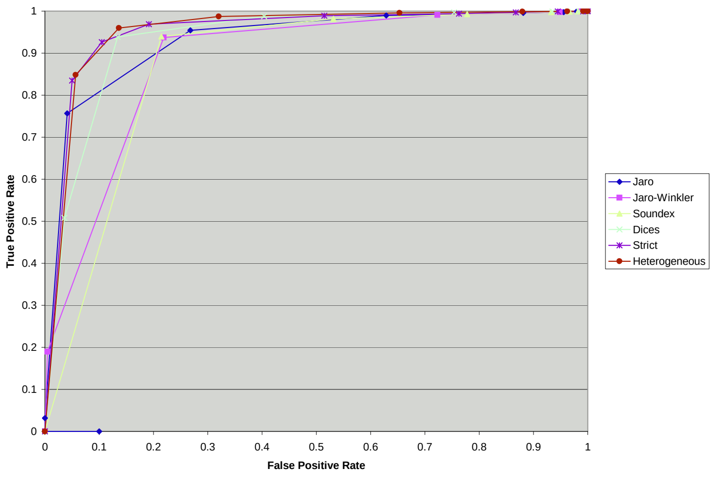

Figure 3.2: ROC Curves for Metrics

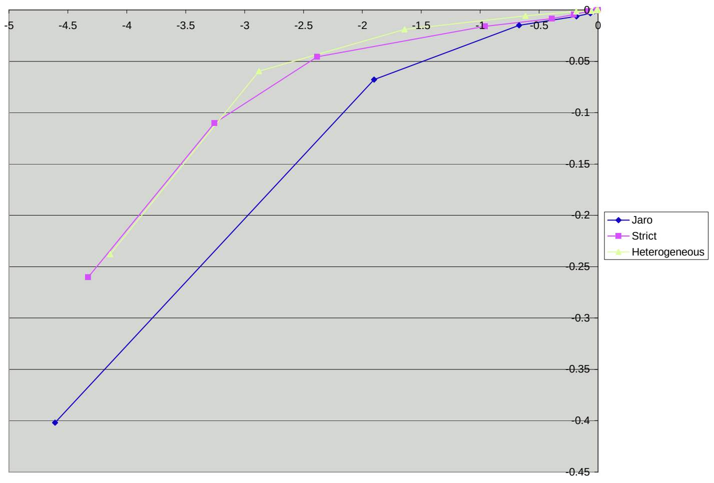

Figure 3.3: *Log* of Top 3 Metrics

Attribute Type	**Metric**
Gender	Binary Discrimination
Name	Soundex
Location	Jaro
Day	1-norm
Month	Dice
Year	1-norm

Table 3.1: Metric Selection Table

ROC (Receiver Operating Characteristic) analysis has its origin in signal detection theory. It was recently introduced in the field of Machine Learning/Data Mining (e.g., see [14]). ROC curves display the true positive rate or sensitivity (i.e., the ratio of the number of matches found to the total number of matches) versus the false positive rate or 1 minus specificity (i.e., the ratio of the number of incorrectly found matches to the total number of mismatches). Each point on the curve represents a decrease in the threshold value used to classify matches. At the far left, the threshold is set to 100%, therefore nothing is classified as a match and thus we get 0% on both the false positive rate and the true positive rate. At the far right, the inverse is true. The threshold is set to its lowest value, so that everything is classified as a match. A steep slope shows that the metric is predicting many instances correctly with few misses. A steeper slope indicates a more accurate system. The other item of importance in ROC analysis is the area under the curve (AUC). A large area suggests that the metric gets more right with a higher threshold.

The strict and the Jaro metric both have slightly better slope than our composite metric (labeled Heterogeneous in Figures 3.2 and 3.3), but our AUC is slightly superior to that of Jaro and much larger than that of the strict metric. It is interesting to note that the strict or binary discrimination metric did very well. This is due to the fact that the experimental data has been cleansed and standardized. An increase in misspellings and missing data would surely decrease its performance, and in turn widen the gap between our composite metric and all other homogeneous metrics.

For comparison purposes with future extensions, a single precision and recall value is calculated with the acceptance threshold set to 0.7, as this gives good results. Precision for a class C, is calculated as the ratio of the number of instances correctly identified as C to the total number of instances classified. Recall is also a ratio, but of the number of instances correctly classified to the number of instances of class C. Precision and recall can be formally defined as:

$$P = \frac{TP}{TP + FP} * 100\%$$

and

$$R = \frac{TP}{TP + FN} * 100\%$$

where TP, FP, and FN refer to the number of true positive, the number of false positive, and the number of false negative predictions, respectively. The final results are shown in Table 3.2, where Distribution denotes the ratio of matches to mismatches in the data and Generations denotes the number of ancestor (respectively, descendant) generations used in the pedigree.

Table 3.2: Version 0.1 Results

	Version 0.1
Distribution	1:2.8
Generations	4 up and 4 down
Precision	71.8%
Recall	94.6%

3.2 Remaining Challenges

Although the above results are encouraging, two main challenges remain.

1. The similarity metric is applied uniformly across the data, i.e., it is non-weighted, and

2. Machine learning algorithms often have difficultly predicting a minority class when the data distribution is highly skewed.

Weights are needed to vary the impact an attribute carries. Version 0.1 treats each attribute equally. Therefore each attribute contributes equally in determining the class of the instance. This is also true across relatives. Thus, the closeness of the date of birth for two individuals carries the same weight, or importance, as their great-grandmother's christening date. A more appropriate method would involve different weights depending on the attribute as well on the relatives being measured.

Further, the system, not human intuition, should determine how much weight ought to be given to each attribute. This allows new models to be created quickly and for insight to be gained on the role of certain attributes in record linkage.

One of the inherent characteristics of record linkage data sets is their (sometimes extreme) skewness, arising from the fact that there are far more mismatches than there are matches, i.e., the probability that two records taken at (almost) random represent the same person is — in practice, at least — very small. Skewness is a major challenge for any learning algorithm and several solutions have been proposed (e.g., see [15]).

Our solution to both of these problems is addressed with a Filtered Structured Neural Network (MAL4:6, version 1.0) and detailed in the following chapter.

Chapter 4

Filtered Structured Neural Network

A structured neural network provides a way for the system to learn and adjust weights based on attributes and relatives. Filtering or blocking methods are used to reduce the effect of skewness. We begin with some additional definitions specific to the context of pedigree-based record linkage.

4.1 Pedigree Definitions

The following definitions extend our earlier set to take into account pedigree. They also form the basis of the design of our structured neural network.

We denote by $R = \{R_0, R_1, ..., R_m\}$ the set of functions mapping individuals to individuals that capture relevant family relationships (e.g., Father(x)). By construction, $R_i(x)$ is a member of the pedigree belonging to individual x. By convention, R_0 is the identity function, i.e., $R_0(x) = x$. For example, a person x may have the relations $\{x, Father(x), Mother(x), Spouse(x), Father(Father(x))\}$.

An extended individual, EI, is an individual together with its relatives up to *uplimit* generations and down to *downlimit* generations.

$$EI(x, uplimit, downlimit) = \{R_0(x), R_1(x), ..R_p(x)\} \tag{4.1}$$

where the R_i refer to the relatives in the pedigree of x within *uplimit* and *downlimit* generations.

Finally, we define the similarity measure between two extended individuals as follows:

$$sim_{EI}\left(EI(x,up,down),EI(y,up,down)\right) = \sum_{i=0}^{|EI(x,up,down)|} \omega_i sim_I(R_i(x),R_i(y)) \quad (4.2)$$

where ω_i is the weight associated with a relative, R_i, and $R_i(x)$ and $R_i(y)$ always refer to the same relative in both pedigrees (e.g., mother). A neutal score (.5) is given if either individual is missing from their respective pedigree.

4.2 Learning Weights with a Structured Neural Network

There are many algorithms and machine learning approaches which could be used to determine weights. A neural network system was chosen as a foundation because of its success in other applications and its ability to fine tune weights. However, there are two major drawbacks in using a standard neural network. The first is that the training time can be very long. A fully connected neural network may contain many hidden layers and nodes which all must be updated after each instance. The second disadvantage to neural networks is that the network is not comprehensible. In a semi-large network, the intricate interdependencies among nodes make it generally impossible for a human to make sense of the weights.

We side-step the first disadvantage by assuming that the accuracy of the system is more important to the success of MAL4:6 than is its training time. Furthermore, once trained, neural networks are very computationally efficient. We overcome the second limitation by imposing structure to the neural network used. A structured neural network is a neural network whose architecture or topology is constrained in some way to bias its learning (e.g., see [16, 17]).

Consider Equation 4.3, an extension of Equations 2.1 and 4.2, where each relative has a different weight (ω_i) and each attribute is also weighted (α_{ij}) differently for each relative.

$$sim(x,y) = \sum_{i=0}^{p} \omega_i \sum_{j=1}^{n} \alpha_{ij} sim_{\gamma_j}(\gamma_j^{R_i(x)}, \gamma_j^{R_i(y)}) \quad (4.3)$$

By designing a network that implements Equation 4.3, the model becomes human readable and facilitates knowledge extraction. Figure 4.1 gives a visual representation of the corresponding network.

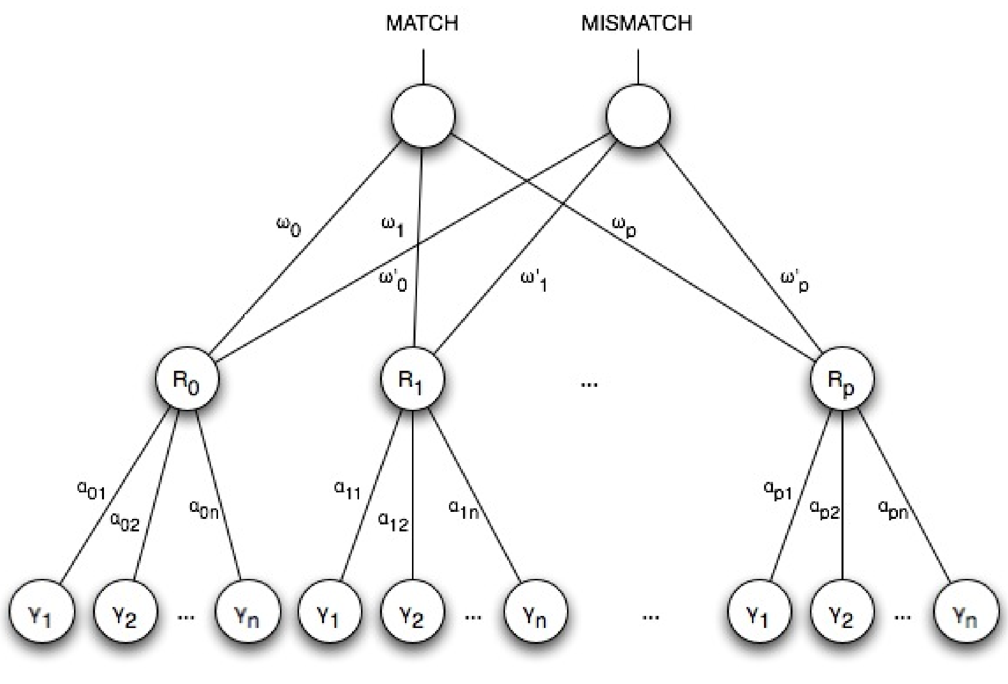

Figure 4.1: A Structured Neural Network

The links in the figure represent weights. Each attribute (e.g., date of birth) has a link to the corresponding relative (e.g., mother). There is also a link between each person in the network and the output nodes. This architecture allows the system to treat each portion of the pedigree (including individual components) separately. Two output nodes allows one set of weights (ω) to focus on predicting a match and another set to focus on predicting a mismatch. This was found to give better results than using a single output node in preliminary experiments. MAL4:6 uses the results of the two output nodes to compute a ratio, which in turn serves to compute final prediction of the pairs' matching status.

It is clear that a structured neural network does not contain the same number of connections as a fully connected neural network. Our structured neural network only contains one hidden layer and each element in the input layer is connected to only one hidden node. The hidden nodes are fully connected to the output nodes. Given i input nodes, h hidden nodes, and p output nodes, a fully connected neural network will contain $h(i + p)$ weights. By contrast a structured neural network of this variety will only contain $(h * p) + i$. Obviously, for large values of i the difference can be very significant. By using all the attributes that exist for all members in a pedigree, there are often hundreds of inputs into the network. The network becomes more readable by reducing the number of weights and connections. While not critical to the success of MAL4:6, the training time is also shortened by reducing the number of updates (a factor of h at the input layer).

4.3 Filtering Neural Networks

Filtering, or blocking, is the process of eliminating as many instances of the over-represented class as possible, thus reducing skewness and easing learning [18]. Blocking is typically applied to highly skewed sets as a preprocessing step. A similar process to the one described in Section 2.2.3 is used here, with a few alterations: a smaller set of rules, fewer attributes (usually name, gender, and date of birth), and a relaxation of δ. The goal is to compare all of the instances against each other (N^2 comparisons) quickly without eliminating any potential matching pairs. After the blocking sequence is completed a new set of size n is formed, where ideally $n << N$. A more thorough approach is performed on the n remaining instances (n^2 operations). MAL4:6 extends this idea by applying a type of filtering method on previously blocked data.

In the spirit of [19], our filtering approach allows a kind of successive re-proportioning of the data through delegation among structured neural networks. Extreme instances, which contribute to the skewness are filtered out at each stage, as follows.

Let $T = M \cup m$ be the current training set, where M is the set of pairs from the over-represented, or majority, class in T (i.e., either match or mismatch) and m

the set of pairs from the other class. Define:

$$\delta_M = \frac{1}{|M|} \sum_{p \in M} \frac{MATCH(p)}{MISMATCH(p)} \tag{4.4}$$

where $MATCH(p)$ is the value of the output node MATCH when p is presented to the network and $MISMATCH(p)$ is the value of the output node MISMATCH. Similarly,

$$\delta_m = \frac{1}{|m|} \sum_{p \in m} \frac{MATCH(p)}{MISMATCH(p)} \tag{4.5}$$

Finally, the ratio for the pair p is defined:

$$r = \frac{MATCH(p)}{MISMATCH(p)} \tag{4.6}$$

Figure 4.2 gives a description of the filtering process. An instance p is classified as the majority class (as determined by T) and filtered out (i.e., removed from the training set) if r lies extreme to δ_M. When the majority class M is match, a value of r above δ_M would be considered extreme. When M is mismatch, a value of r below δ_M is considered extreme. A new training is built from the unfiltered instances. This is to allow a new network to be built based on data with a more even distribution. The process is repeated with the new structured neural network.

As instances from the majority class are removed, the distribution for both the training and test set becomes more balanced (closer to 1:1) if the majority class of the test set is the same as the majority class in the training set. This allows future networks to focus their learning time on the remaining instances without the influence from the outliers. The basic assumption is that instances with highly skewed scores towards a classification will not benefit from additional learning. In fact, their presence may hinder the ability of the network to correctly distinguish other instances.

The final epoch of the last filtering stage is similar to the earlier stages, except a threshold of the minority class is also computed, δ_m. If the minority class is assumed to be a match, instances with ratios above δ_m would be classified as a match and all remaining instances are classified as mismatches (the majority class of the training set). Precision and recall values are computed based on the number of true/false positive/negative obtained in all networks used.

```
function Filtering($trainSet$, $testSet$, $numTP$, $numFP$, $numTN$, $numFN$, $epochsLeft$)
Initialize a separate neural network
Using back propagation, train the network using instances in $trainSet$ for all training cycles
$\delta_{mismatch} = \frac{1}{|mismatch|} \sum_{p \in m} \frac{MATCH(p)}{MISMATCH(p)}$

$\delta_{match} = \frac{1}{|match|} \sum_{p \in M} \frac{MATCH(p)}{MISMATCH(p)}$

Loop through each pair $p$, in $trainSet$ and $testSet$
Compute $r_p = \frac{MATCH(p)}{MISMATCH(p)}$

If $|mismatch| > |match|$ in $trainSet$
  If $r_p \geq \delta_{mismatch}$ AND $p \in trainSet$
    Insert $p$ into $newTrainSet$
  If $r_p < \delta_{mismatch}$ AND $p \in testSet$
    Classify $p$ as mismatch and increment $numTP$, $numFP$, $numTN$, $numFN$ as appropiate
  Else if $epochsLeft = 0$
    If $r_p < \delta_{match}$ AND $p \in testSet$
      Classify $p$ as mismatch and increment $numTP$, $numFP$, $numTN$, $numFN$ as appropiate
    Else if $p \in testSet$
      Classify $p$ as match and increment $numTP$, $numFP$, $numTN$, $numFN$ as appropiate
Else
  If $r_p \leq \delta_{match}$ AND $p \in trainSet$
    Insert $p$ into $newTrainSet$
  If $r_p > \delta_{match}$ AND $p \in testSet$
    Classify $p$ as match and increment $numTP$, $numFP$, $numTN$, $numFN$ as appropiate
  Else if $epochsLeft = 0$
    If $r_p > \delta_{mismatch}$ AND $p \in testSet$
      Classify $p$ as match and increment $numTP$, $numFP$, $numTN$, $numFN$ as appropiate
    Else if $p \in testSet$
      Classify $p$ as mismatch and increment $numTP$, $numFP$, $numTN$, $numFN$ as appropiate
If $p$ is not classifed AND $p \in testSet$
  Insert $p$ into $newTestSet$
If $epochsLeft > 0$
  Call Filtering($newTrainSet$, $newTestSet$, $numTP$, $numFP$,
    $numTN$, $numFN$, $epochsLeft$-1)
```

Figure 4.2: Filtering Algorithm

Chapter 5

Experimental Results

A set of experiments are presented to measure the effectiveness of filtering structured neural networks. The experiments are quantified in terms of precision and recall.

5.1 Data

The data used in the experiments presented was provided by the Family and Church History Department (FCHD) of The Church of Jesus Christ of Latter-day Saints. The data consists of 17,350 pairs and their associated pedigrees. Due to legal concerns over privacy, the vast majority of the data is from the 1500-1700's.

The instances were randomly divided into 3 sets for experimental purposes. The first two sets were used for development and training of MAL4:6. When development was completed, MAL4:6 was tested using the third set. All results in this chapter were obtained using this setup.

The data was initially pre-processed using a blocking algorithm (obvious non-matches are eliminated) performed by the FCHD. Thus all remaining instances are considered close. Each instance was classified by hand by a professional genealogist. The final distribution between matches and non-matches is 1:2.8 (i.e., 2.8 mismatches for each match).

5.2 Verification of Filtering and Structured Neural Networks

For experimental purposes, we created data sets with distributions of 1:1 and 1:100. Each is an extreme case, 1:1 being most desirable and 1:100 being least desirable. We note at the onset that the data set with 1:100 distribution, because it was generated from post-blocking data, provides a kind of worst-case scenario in terms of performance since all pairs are close, whether they are matches or mismatches. In a typical case, blocking is applied on skewed but more diverse data. The 1:1 data set is also used to test how filtering affects a balanced data set.

Tables 5.1 and 5.2 report the results of using filtering versus no filtering. Table 5.1 shows how the distributions are affected at each filtering pass. Table 5.2 shows the precision and recall values if the current network were the last network. The non-filtering approach was trained with 1,000 epochs (or training cycles) and each filtering pass consisted of 200 epochs.

Table 5.1: Distribution Shifts Under Filtering

Dis.	**Pass 1**	**Pass 2**	**Pass 3**	**Pass 4**	**Pass 5**
1:100	1:79.7	1:28.9	1:3.18	–	–
1:1	1:0.42	1:4.45	1:2.59	1:1.42	1:2.47

Table 5.2: Precision/Recall Evolution With Filtering

Dis.	**None**	**Pass 1**	**Pass 2**	**Pass 3**	**Pass 4**	**Pass 5**
1:100	25.0/33.3	70.0/33.3	44.4/85.7	44.4/85.7	–	–
1:1	80.3/81.6	91.6/85.7	91.4/86.7	88.0/94.0	88.6/93.5	88.9/93.8

The 1:100 distribution went through only three passes, since after the third

pass there were not enough match elements left in the training set for the system to learn properly. It is interesting to note that the distribution becomes less balanced between the fourth and fifth pass on the 1:1 distribution. This is because the training set had a different majority class than the testing set at that stage. While the training set became more balanced, the test set did not. This appears not to have impacted the learner as precision and recall actually improve between these stages.

Table 5.2 shows an interesting property of precision and recall values. Typically one value increases and the other decreases. Either value can be easily increased simply by decreasing the other value. For instance, if recall is desired, the threshold can be set low as to declare match on more instances. This has the desired effect, but it may decrease precision as more instances are misclassified. An increase in both precision and recall demonstrates improved accuracy in the system.

Both distributions seem to benefit from filtering. The 1:100 shows an increase of 19.4% in precision and over 50% in recall. After the first pass, both precision and recall values are increased with the 1:1 distribution. Overall, between the second pass and the last pass, recall increases by 8.1% with a small decrease of 2.7% in precision. This performance may be acceptable depending on the user's priorities. Traditionally, recall is favored when dealing with highly skewed data and precision becomes increasingly more important as the distribution is balanced.

Finally, Table 5.3 compares the results of the extension presented here with the results of version 0.1 (see Chapter 3) of the system. Because of the sparseness of the dataset, using descendants did not seem to offer much benefit and was therefore excluded from experiments with Version 1.0.

Table 5.3: Version 0.1 vs Version 1.0

	Version 0.1	**Version 1.0**
Distribution	1:2.8	1:2.8
Generations	4 up and 4 down	3 up
Precision	71.8%	85.2%
Recall	94.6%	90.9%

Precision was increased by over 13.4%, while recall decreased by only 3.7%. A decrease in recall can be expected, since at each stage of the filtering process some pairs are misclassified and not introduced into the next stage. We believe that this reduction of recall is offset by the significant increase in precision and overall accuracy of the system. These results validate the approach of weights and filtering. Additionally, they compare well with results generally reported in the record linkage literature(e.g., see [20, 21, 22]).

5.3 Interpreting Structured Neural Networks

As mentioned earlier, a major benefit of structured neural networks is their readability by a human. The weights from different networks can be compared against each other. This allows researchers to identify attributes which may change in importance. As an illustration, we compare the first networks from the 1:100 and 1:1 distributions of the experiments in Section 5. Attributes which have a high change in weight values suggest that as the distribution changes, these attributes either increase of decrease in importance. A small sample of attributes with the weights used in the different networks are displayed in Table 5.4.

Table 5.4: Weights of Attributes/Relatives for Two Distributions

Attribute/Relative	**1:1 Ratio**	**1:100 Ratio**	**Difference**
Self's Christening Loc.	-70.0	8.84	-78.84
Father's Given Name	-8.22	1.24	-9.46
Paternal Grandmother's Marriage Loc.	-0.159	-0.171	.012
Self's Death Year	15.48	-9.95	25.43
Self's Given Name	15.77	-8.15	23.92
Self (Relative)	9.66	-7.85	17.51
Mother (Relative)	-5.94	-.35	-5.59

The knowledge contained in Table 5.4 allows researchers to focus on specific attributes depending on the distribution of the sets. For instance, a genealogist who deals with highly skewed data may consider focusing on obtaining the correct christening location. On the other hand, as the distribution becomes more balanced, the expert may want to rely more on other attributes at the individual's level (e.g., death year). Some attributes, such as the paternal grandmother's marriage location, appear to play little role in both distributions. This knowledge can also be helpful to researchers who use rule-based approaches. A researcher in this field can eliminate time in obtaining or building rules not focusing attention on this seemingly unimportant attribute. The weights on relatives also vary depending on the distribution. More emphasis is placed on the individual's self node for more balanced distributions, while the mother's node actually decreasing in importance. Reasons for these fluctuations can be researched further and exploited by a human expert.

5.4 Utilizing Pedigree Data

One of the main motivations for MAL4:6 is the ability to use pedigree information to improve record linkage, especially in the case of sparse individual data. As a first attempt at measuring the improvement, we run MAL4:6 on the original data set with and without pedigree data. When no pedigree data is used, the structured neural network consists of only one hidden node, which represents the individual. The results are shown in Table 5.5.

Table 5.5: Comparison of Accuracy With and Without Pedigree Data

	No Pedigree	**Pedigree**
Distribution	1:2.8	1:2.8
Generations	Individual only	3 up
Precision (Match)	85.3%	85.2%
Recall (Match)	89.0%	90.9%
Precision (Non-Match)	95.9%	96.5%
Recall (Non-Match)	94.2%	94.1%

While recall is improved slightly, the accuracy measurements are rather close.

This experiment alone does not seem to provide a strong argument for using pedigree data. An explanation for this somewhat unexpected behavior can be found by closely examining the training data.

5.4.1 Data Characteristics

The FCHD's record linkage method relies on hand-crafted rules that focus on individual data. With only a few exceptions, pedigree data is not used in these rules, and was therefore likely not a priority in the FCHD's data collection strategies. Furthermore, as mentioned in Section 5.1, the majority of the data provided by the FHCD is from the 1500-1700's. Pedigree data becomes increasingly scarce as pedigrees go further back in time.

Consequently, most of the instances in the datasets provided by the FCHD contain large amounts of data at the individual level, but very little data at the pedigree level. In fact, out of the 17,350 pairs, only 1005 (8.9%) have at least one corresponding relative, meaning they shared at least one relative at the same pedigree position.

Missing even a single relative can have a major impact on a pedigree-based system such as MAL4:6. A missing father means that all of his ancestors are also missing. It is difficult for neural networks to learn and adjust values appropriately when the training set is sparse. This is due in part to the way neural networks adjust the weights and the necessary choice of input values for missing data. In an attempt to remain unbiased in the context of inputs lying in $[0, 1]$, a value of 0.5 is inserted for each missing value in our implementation. Weights are adjusted after each iteration, even if the attribute is missing from either individual in p. Hence, weights on missing value links are also modified.

When no pedigree data is used, the network contains 40 input nodes (i.e., 40 attributes). On average, 26.6% of these nodes have data associated with them. In contrast, when three generations upward are used in the experiments, a total of 880 inputs exist. Of these, on average only 1.6% have data associated with them. The system has difficultly learning because there are an overwhelming number of network nodes containing missing data. Hence, the observed improvement is only marginal in this case. However, the system gains in generality since it needs not be re-designed for each set of possible attributes.

5.4.2 Situations Targeted by MAL4:6

Genealogy work is typically done based on a pedigree tree. New individuals are discovered through a process of finding a known person's relative. For example, children, parents, and/or spouses are discovered for a person already in the pedigree. Little information is known about this new individual upon discovery. Generally a name and a gender can be inferred. Standard record linkage approaches may have difficulty in this environment, as they may require further attributes, such as dates and locations before accurate linkage could take place. On the other hand, MAL4:6 is designed specifically for these types of situations. This is demonstrated in the following experiments.

In a first experiment, we exploit the data available from the FCHD by restricting attention to instances that have a lot of missing data at the individual level but a low amount of missing data at the pedigree level. As expected, only a very small number of instances fit this criterion (≈ 60 instances). These instances used the structured neural network but no filtering was done due to the small number. The results are displayed in Table 5.6.

Table 5.6: Instances with Rich Pedigrees

	No Pedigree	**Pedigree**
Distribution	1:2.1	1:2.1
Generations	Individual only	3 up
Precision (Match)	73.7%	78.9%
Recall (Match)	100.0%	100.0%
Precision (Non-Match)	100.0%	100.0%
Recall (Non-Match)	66.7%	71.4%

While no claims should be made with such a small data set, it is promising that significant improvements are observed. To further test this effect, we conducted the following experiment.

We consider all of the instances provided by the FCHD. However, we modify them systematically to reduce information at the individual's level, whilst retaining pedigree information. All attributes on the individual's level are removed except for first name, last name, and gender. All of the available information about the other members of the pedigree remains intact. This scenario might illustrate the situation of discovering a new child for a known individual. No information about the child is known except for the name, gender, and placement in the pedigree of the new person.

Filtering is applied. Precision and recall values were computed after five filtering passes. They are reported in Table 5.7.

Table 5.7: Little Individual Information

	No Pedigree	**Pedigree**
Distribution	1:2.8	1:2.8
Generations	Individual only	3 up
Precision (Match)	58.0%	64.0%
Recall (Match)	95.0%	95.3%
Precision (Non-Match)	97.5%	97.9%
Recall (Non-Match)	74.7%	80.0%

The precision is increased by 6.0%, when additional pedigree information is used. In addition, no decrease in recall is experienced by the system, suggesting that using pedigree data in this situation is indeed beneficial. Larger, more realistic datasets with high amounts of pedigree information should confirm these results.

5.5 Comparison with FCHD Standard

We conclude this chapter with a partial comparison with the current FCHD approach to record linkage. The FCHD results we were provided are based on the combination of a system of elaborate human-crafted rules (as discussed in 2.2.3) with a support vector machine. We note at the onset that, concurrent and independent of our work, the FCHD has been increasingly incorporating pedigree information in its rules.

With the current, individual-oriented datasets, the FCHD's best reported results are 96.8% precision for 96.8% recall. This is clearly, and not surprisingly so, far superior to MAL4:6 results under the same conditions. As mentioned earlier, this may be due in part to the human expertise embedded in the rules and their subsequent fine-tuning to the data by hand.

A more meaningful comparison here would involve running the FCHD's approach and MAL4:6 side-by-side on some of MAL4:6's targeted situations. Unfortunately, it has proven difficult for the FCHD to apply its approach to the experiments designed in section 5.4.2. Rather than not try at all, however, we resorted to designing experiments as reasonable as possible to allow some level of comparison between MAL4:6 and the FCHD's approach. These results are to be taken with caution since neither corresponds to the area of expertise of either system.

Both experiments focus on the situation where little information is known about the individual being linked. In the first experiment, only the attributes corresponding to the name components for an individual are used as inputs into the neural networks. In the second experiment, we extend this to including the name components of relatives within one generation upward (i.e., mother and father). Filtering was further turned off in MAL4:6. While this does negatively impact the results, it does allow the construction of precision/recall curves. Figures 5.1 and 5.2 show the results.

Precision/recall graphs make it possible to visualize the trade-off between the two quantities as the similarity threshold varies. Figure 5.1 shows that with only the name, the FCHD's approach is able to achieve rather high performance (74% precision with 94.5% recall). This further highlights the fact that the available data is very rich in names, which, in turn, seem to be a key component in the rule-based approach. MAL4:6 achieves 60% precision at the same recall point. The FCHD's

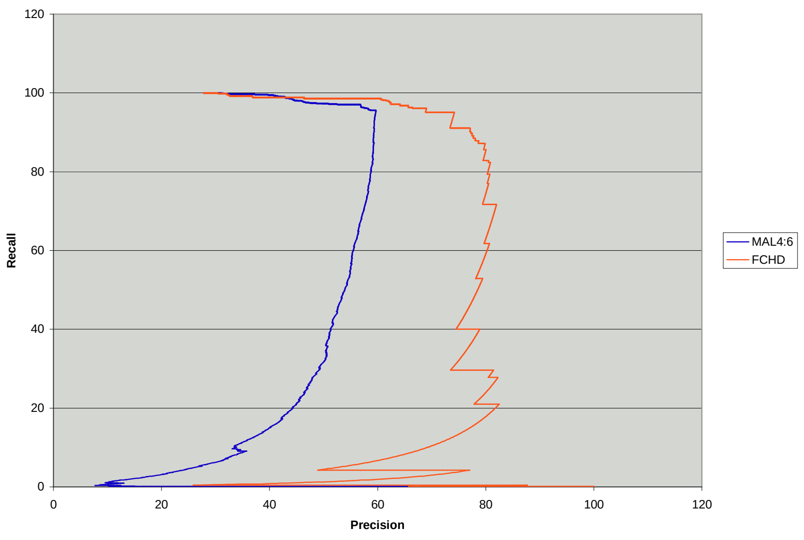

Figure 5.1: Only the Individual's Name Used

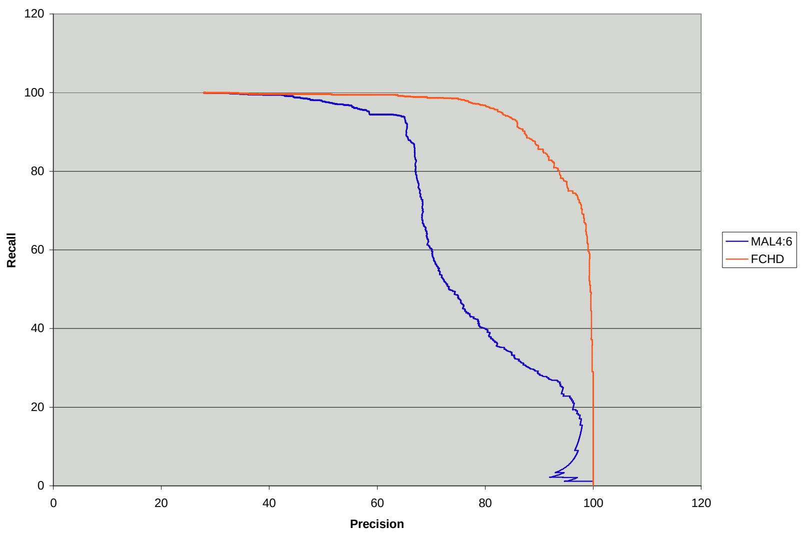

Figure 5.2: Individual and Near Relatives' Name Used

approach benefits more than MAL4:6 does when adding the names of close relatives. Figure 5.2 shows that at 94.5% recall point, the FCHD has precision up to 84% while MAL4:6 reaches only 65%. Although likely an artifact of the rules used, this may also be an indication that either MAL4:6's name similarity metric could be improved or that using raw similarity scores for names may be more detrimental than expected.

As a final note, we remark that, as an approach, MAL4:6 does offer advantages in terms of versatility and generality. MAL4:6 is completely automatic. It does not rely on human experts to alter (modify, add or delete) rules or adjust inputs into the network. This may become particularly relevant as new datasets are encountered. Datasets with genealogies from different time periods and/or different locations are likely to exhibit different characteristics than those in the FCHD's current dataset. For instance, date of birth is a common attribute that exists in the tested datasets. However, it may not be common in pedigrees from the South Pacific. Other attributes, such as tribe status may be more important to this culture and thus recorded in the pedigree. A rule-based system would need to make changes in its rules to reflect these characteristics. Other rules besides date of birth would need to be modified as well to compensate for the missing values. Additionally, new attributes would result in new rules being created, tested and modified. This process could be very time consuming and may need to be repeated for several time periods and locations. MAL4:6 would only need minor alterations to the network structure (if new attributes are used) or no transformations at all. Its learning mechanism would automatically take care of the various degrees of importance (i.e., weights) of attributes. We believe that this generality is desirable and will pay off in the long run.

Chapter 6

Conclusion

Current record linkage methods focus on individual data being richly populated. This is often the case when dealing with data which has been carefully examined and researched by genealogy experts for use in research. There are, however, circumstances when a researcher may want to link an individual even if little is known about them, but more is known about their genealogy. By linking to an already known entity, a researcher is able to glean new information and apply it to the recently discovered individual. Table 5.7 does indeed suggest that by applying the full pedigree, precision may be increased without losing performance in recall.

6.1 Contributions

MAL4:6 offers insight into how pedigree information can be used in the record linkage process. Often ignored, pedigree information is valuable and may become essential as individuals conduct merging algorithms on the fly over heterogeneous sources of data. MAL4:6 also demonstrates how knowledge from a structured neural network can be compared and examined with other structured neural networks of the same variety. This can provide insight into areas where further research might be necessary. Weight fluctuations between networks can give an indication of the importance of an attribute or a relative.

6.2 Future Work

Further experiments are needed to more fully validate MAL4:6's ability to improve record linkage through pedigree information. Obtaining records with high amounts of pedigree information is therefore an area of future work. These records will be created by additional research or be made available through more current records. Improved results can hopefully be obtained as pedigrees become richer in information.

Dealing with missing data is a problem whenever machine learning algorithms are used. Decisions are learned without a full knowledge of the space. A possible solution is to combine attributes at the input layer. This will lessen the amount of missing information. On the other hand, this prohibits the system from learning weights for a specific attribute. Instead, a weight would be learned for a group of attributes. Groupings could be biased to reduce the effects. Birth information (place and dates) which now constitute seven distinct fields could be aggregated into one field, for example.

A recent grant provided by Brigham Young University to the Data Mining Lab will also allow researchers to explore alternative record linkage techniques. One such alternative is Bayesian networks. Bayesian networks may lend themselves well to these problems because they allow for measured probabilities to be applied. Additionally, confidence in a classification can be assigned which may prove useful for a human researcher.

Bibliography

[1] E. Rahm and H. Do, "Data cleaning: Problems and current approaches," *IEEE Data Engineering Bulletin*, vol. 4, pp. 23–34, 2000.

[2] D. Quass and P. Starkey, "Record linkage for genealogical databases," in *Data Cleaning, Record Linkage and Object Consolidation Workshop*, 2003, pp. 40–42.

[3] H. Newcomb, J. Kennedy, S. Axford, and A. James, "Automatic linkage of vital records," *Science*, vol. 130, pp. 854–959, 1959.

[4] I. Fellegi and A. Sunter, "A theory for record linkage," *Journal of the American Statistical Association*, vol. 64, pp. 1183–1210, 1969.

[5] W. Winkler, "Matching and record linkage," *Survey Methods for Business, Farms, and Instituations*, pp. 355–384, 1995.

[6] ——, "Data cleaning methods," in *KDD-2003 Workshop on Data Cleaning, Record Linkage, and Object Identification*, 2003, pp. 1–6.

[7] W. Cohen, P. Ravikumar, and S. Fienberg, "A comparison of string metrics for matching names and records," in *KDD-2003 Workshop on Data Cleaning, Record Linkage, and Object Consolidation*, 2003, pp. 19–24.

[8] M. Jaro, "Advances in record linking methodology as applied to the 1985 census of Tampa Florida," *Journal of the American Statistical Society*, vol. 89, pp. 414–420, 1989.

[9] ——, "Probabilistic linkage of large public health data files," *Statistics in Medicine*, vol. 14, pp. 491–498, 1995.

[10] W. Winkler, “The state of record linkage and current research problems,” Statistical Research Division, U.S. Bureau of the Census, Washington, DC, Tech. Rep., 1999.

[11] J. Zobel and P. Dart, “Finding approximate matches in large lexicons,” *Software–Practice and Experience*, vol. 1, pp. 331–345, 1995.

[12] L. Dice, “Measures of the amount of ecologic association between species,” *Ecology*, pp. 297–302, 1945.

[13] T. Sorensen, “A method of establishing groups of equal amplitude in plant sociology based on similarity of species content and its application to analysis of the vegetation on danish commons,” *Biologiske Skrifter*, vol. 5, pp. 1–34, 1948.

[14] F. Provost and T. Fawcett, “Robust classification for imprecise environments,” *Machine Learning*, vol. 42, pp. 203–231, 2001.

[15] B. Zadrozny, “Learning and evaluating classifiers under sample selection bias,” in *Twenty-first International Conference on Machine Learning*, 2004, pp. 903–910.

[16] G. Lendaris, M. Zwick, and K. Mathia, “On matching ann structure to problem domain structure,” in *Proceedings of the World Congress on Neural Networks (WCNN’93)*, 1993, pp. 488–493.

[17] C. Lee, T. Rey, J. Mentele, and M. Garver, “Structured neural network techniques for modeling loyalty and profitability,” in *Proceedings of SAS User Group International (SUGI 30)*, 2005, pp. 082–30.

[18] L. Gu and R. Baxter, “Adaptive filtering for efficient record linkage,” in *Fourth SIAM International Conference on Data Mining*, 2004, pp. 477–481.

[19] C. Ferri, P. Flach, and J. Hernandez-Orallo, “Delegating classifiers,” in *Proceedings of the Twenty-first International Conference on Machine Learning (ICML’04)*, 2004, pp. 289–296.

[20] P. Singla and P. Domingos, “Multi-relational record linkage,” in *KDD-2004 Workshop on Multi-Relational Data Mining*, 2004, pp. 31–48.

[21] W. Winkler, “Machine learning, information retrieval and record linkage,” in *Section on Survey Research Methods*, 2000, pp. 20–29.

[22] P. Christen and K. Goiser, “Towards automated data linkage and deduplication,” 2005, submitted. (Available at http://datamining.anu.edu.au/publications/2005/pakdd2005-automated.pdf).