Public Record Aggregation Using Semi-supervised Entity Resolution

Jack G. Conrad, Christopher Dozier, Hugo Molina-Salgado, Merine Thomas, Sriharsha Veeramachaneni
Thomson Reuters Research and Development
610 Opperman Drive
Saint Paul, Minnesota 55123 USA
{jack.g.conrad, chris.dozier}@thomsonreuters.com

ABSTRACT

This paper describes a highly scalable state of the art record aggregation system and the backbone infrastructure developed to support it. The system, called PeopleMap, allows legal professionals to effectively and efficiently explore a broad spectrum of public records databases by way of a single person-centric search. The backbone support system, called Concord, is a toolkit that allows developers to economically create record resolution solutions. The PeopleMap system is capable of linking billions of public records to a master data set consisting of hundreds of millions of person records. It was constructed using successive applications of Concord to link disparate public record data sets to a central person authority file. To our knowledge, the PeopleMap system is the largest of its kind. In contrast, the Concord support system is a novel record linkage tool that uses a new semi-supervised training technique called 'surrogate learning' to enable the rapid development of record resolution solutions.

Categories and Subject Descriptors

H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval—*clustering*; H.3.4 [**Information Storage and Retrieval**]: Systems and Software—*Performance evaluation (efficiency and effectiveness)*

General Terms

Design, Experimentation, Performance, Evaluation

Keywords

named entity extraction, record linkage, record matching

1. INTRODUCTION

Given the growing role that electronic public records play today, it is absolutely essential for legal professionals to be able to access and search them in an efficient and cost effective manner. In order to do this in large societies like the U.S., challenges of accuracy and speed in addition to scale need to be addressed. Databases consisting of billions of records are no longer viewed as exceptional. It is an increasingly complex and daunting task given the extent of today's public records space. To be able to enable access and search of such repositories, the task of gathering and aggregating such records into useful person-centric sets becomes a key processing hurdle for legal information service providers.

This paper describes a record linkage system known as PeopleMap that legal and law enforcement personnel use to search databases containing public records of particular people. The system contains hundreds of millions of person records, a number comparable to the population of the United States, and attaches to them several billion public records. The records in question come from a broad spectrum of public records databases. Such records are indispensable for lawyers and others serving as litigators, private investigators, government employees as well as compliance professionals in the corporate arena. The system facilitates accurate and comprehensive public records research.

PeopleMap allows legal professionals to uncover marriage and divorce records, professional licenses, financial investments, real estate transactions, criminal records and other public records that pertain to a single individual. The system is effective because it leverages all of the data elements available in a record, rather than those included in a query alone. The practical utility and primary contribution of the system results from its efficiency: formerly investigators might have to search 20 to 40 public records databases for a comprehensive public records inquiry, whereas PeopleMap enables a single search to assemble all of the records of interest (Figure 1). The time and cost savings from such aggregation can be dramatic. No other statistic-based linking system that we are aware of has been applied to a record linkage problem of this scope in the public records domain.

Record resolution systems (RRS) form the heart of the PeopleMap system. In addition to describing PeopleMap, in this paper we will describe Concord, the record resolution tool we created to rapidly produce resolution subsystems for the various public record sets we linked to person records. Concord enables software engineers to build Java-based RRSs quickly. It also allows users to interactively configure a RRS by specifying match feature functions, blocking functions, and semi-supervised machine learning methods for a specific resolution problem.

ICAIL '11, June 6-10, 2011, Pittsburgh, PA USA

2. PRIOR WORK

Winkler of the U.S. Census Bureau has laid the foundation for some of the earliest formal research into record linkage [11, 12]. By contrast, Cohen, et al. have performed some of the most thorough research into addressing issues of scale, speed and operationalization for matching public records [6, 5]. Like Cohen, McCallum, et al. have approached the problem as a high-dimensional clustering efficiency problem [8]. By contrast, Verykios, et al. deploy a Bayesian decision model for cost optimal record matching [10]. During the same period, Bhattacharya, et al. show how an iterative approach can yield affirmative results when cleansing and integrating records [3]. More recently, Benjelloun, et al. have developed techniques in a toolkit utility that can be used in a variety of domains and across a diversity of fields for good matching performance [2]. In terms of creating a framework for record matching across records and authority files, Baxter and Christen, et al. have extended this idea by developing an experimental system for performing such repeated tasks [1, 4]. As such, their system is one of the few of its kind that permits users to develop their own parameterized record linkage application. Its deficiency stems from its limited scalability.

3. PEOPLEMAP

A recent addition to the *Westlaw* legal research platform,[1] PeopleMap is a system that allows legal professionals to view public information connected to person records whose cardinality is comparable to the population of the United States. Moreover, it serves as an investigative tool that enables researchers to discover relationships between people, assets and other public records as well as to visualize results. It can support litigation (examining backgrounds of witnesses, parties, or jury candidates), person searches (finding witnesses, heirs, child support sources), and due diligence (probing backgrounds of potential business partners, executives, and past associates).

3.1 Underlying Challenges

PeopleMap relies on the ability to link together all records pertaining to the same person with high precision and recall. In developing the system, it was necessary to create one of the largest person-centric databases in existence. In total, it identifies and maintains several billion relationships. In addition to scalability, the solution to the problem had two primary requirements: accuracy and throughput. The principal metrics tracked were precision, the percentage of current matches correct among those made, and recall, the percentage of correct matches made among all possible matches in the database. The need for high precision within the legal applications it serves is clear. Legal researchers require highly accurate and reliable information. The throughput requirement, by contrast, is motivated by two factors: (1) a very large number of public records needed to be processed before the system was complete, and (2) the large number of on-going updates requiring continuous linking and verification. Given these requirements, the solution that was developed relied on one master database representing the universe of available people and linked records from the remaining databases to their corresponding records in the master database.

[1] www.westlaw.com

3.2 Key Dimensions

It is also worth describing some of the other key dimensions of the PeopleMap system. These include the following. PeopleMap:

- learns how to associate records through all available pieces of evidence such as names, addresses, phone numbers, dates of birth, and other key identifying information such as portions of addresses;
- is optimized to overcome partial and incomplete data, misspellings and errors in the information found in public records data;
- incorporates demographic statistics into the matching process to accurately identify matches based on data such as name rarity in a particular geographic area;
- scores relationships with a confidence level to indicate the strength of the relationship based on all matching information;
- has been tested extensively which has shown the system to be highly accurate and robust against variations in the data quality.

A sample PeopleMap screen shot is shown in Figure 1. The screen shows record sets attached to a person name *Jane Sample-Document*. Note that, for privacy reasons, we are showing fictitious records attached to Jane, a fictitious person.

The types of public records attached to individual person records in PeopleMap include real estate transactions, auto, boat and aircraft ownership records, divorce filings, criminal records, arrest records, stock ownership records, court dockets, and many others.

The initial person records for PeopleMap came from a credit agency that possessed basic name and address information for most of the people in the United States. From this file, we created a PeopleMap master file of person records and assigned a unique GUID to each record. We then proceeded to link when justified by the evidence a wide spectrum of public records to these master records.

Linking a public record to a PeopleMap master record involves extracting person-centric information from the record to form one or more structured person records (called `pubrec` records here) and then linking these records to the PeopleMap master records with a record linkage system. Figure 2 shows the dataflow diagram for the linking process which includes a pair of examples.

A `pubrec` record corresponds to a single individual. In many cases, a single public record document can result in the creation multiple `pubrec` records. For example, a public record involving a real estate transaction might generate at least two `pubrec` person records: one for the seller and one for the buyer.

The underlying record linkage system was provided by Concord (which is described in more detail in Section 4). The similarity functions used for linking a `pubrec` to a person master record depend on the types of populated fields available. These include: Levenshtein string similarity comparison of first names after normalization, Jaro string similarity of last names, Smith Waterman string similarity on street address information [5], customized comparisons for

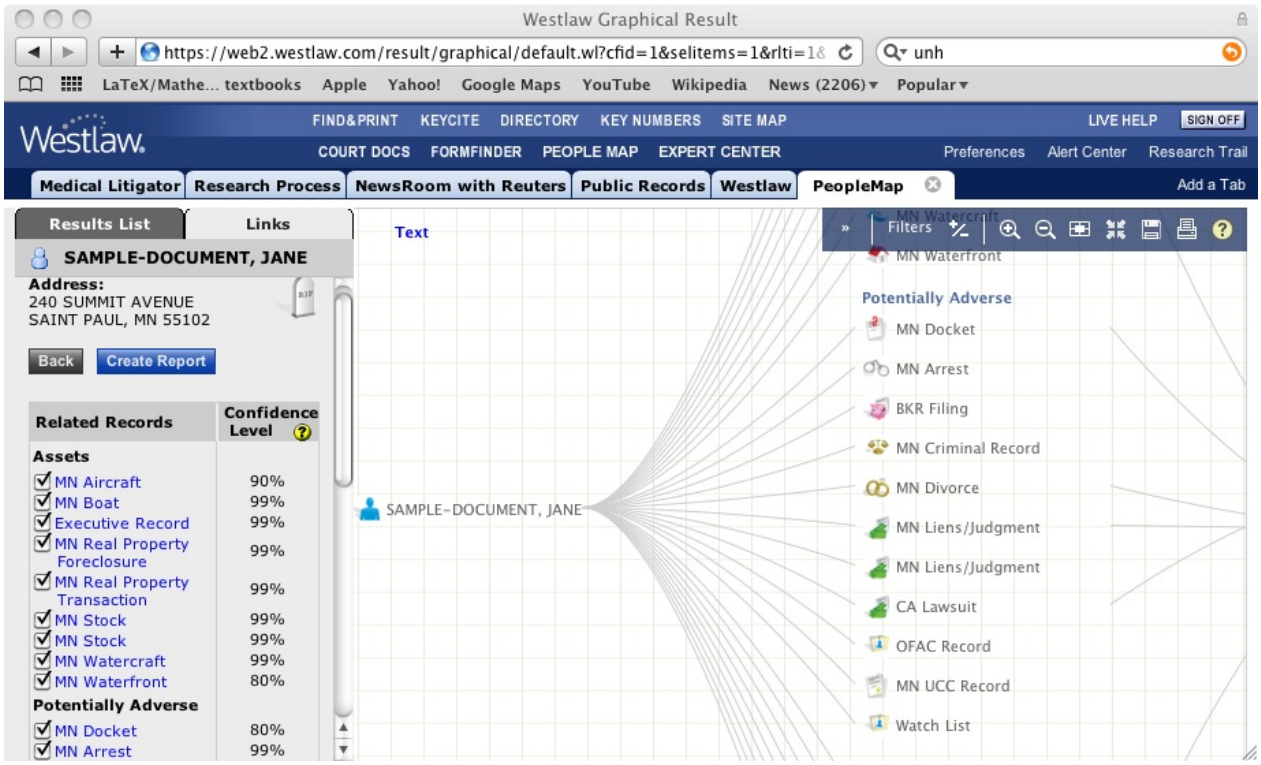

Figure 1: PeopleMap Screen

location information (city, county, state), specialized comparisons for zip code information to account for partial matching, a phone number similarity function, and a date of birth comparison function.

Each `pubrec` record linkage solution consists of two key Concord enabled components, one for blocking and the one for resolution. *Blocking* is concerned with dynamically constructing a sequence of queries (against the master record database) that will retrieve the smallest block (or set) of records that is likely to contain the target record. For example, a Social Security Number query would retrieve a block of size one that contains the target record, while a last name query may retrieve thousands of records. A blocking strategy is concerned with executing more specific queries before less specific ones, and effective blocking criteria are crucial to avoid unnecessary computational expense and ensure high throughput. *Resolution* is concerned with automatically learning and applying a matching function that scores the match probability of a `pubrec` with each master record in a retrieved block. It is worth noting that at the start of the project, we relied upon approximately 10K human-labeled training examples; over time, however, we found that the need for such training data diminished appreciably as the utility of our learning algorithm was realized. In the case of PeopleMap, Concord enabled the learning and application of a match score routine through the use of an original semi-supervised machine learning technique called surrogate learning (described in Sec. 4.5).

4. RECORD LINKAGE USING CONCORD

The Concord backbone was used to create record linkage solutions to connect sets of `pubrecs` extracted from public record documents to person records in the PeopleMap database.

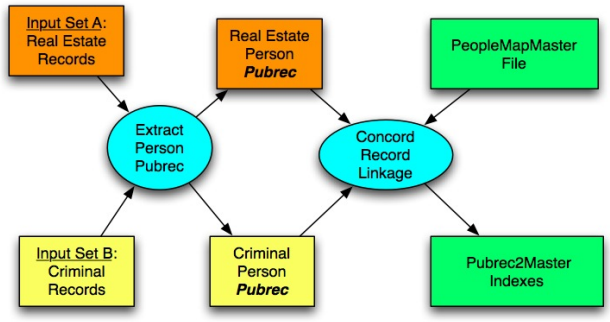

Figure 2: PeopleMap with Concord

We will discuss Concord in the sections following and will provide illustrations using two files containing corporate officers and directors information. Privacy concerns constrain our direct use of general public record information.

4.1 Overview of Concord

Concord provides a quick way to build file record resolution systems. The steps one must follow to create a resolution solution with Concord are described below. For purposes of our discussion, we assume that we wish to match the records in a *source_file* to the records in a *target_file*. In PeopleMap, the *source_file* is a `pubrecs` file and the *target_file* is the person `master file`.

1. **Analyze and align data** in *source_file* and *target_file* records. This means that one must determine which fields in *source_file* records are semantically compatible with fields in *target_file* records. For example, a street address field in a *source_file* may correspond to an address1 field in a *target_file*.

2. **Define feature functions** between fields in *source_file* and *target_file* records. Feature functions are used to determine whether a *source_file* and a *target_file* record pair match. A feature function typically compares one or more fields in a *source_file* record with one or more fields in a *target_file* record and computes a similarity score.

   Concord provides an interactively available library of feature functions that can be selected and attached to field elements in a *source_file* and *target_file* to produce feature function values for inclusion in a feature 'vector.'

3. **Specify a blocking function** that accepts information from a *source_file* record and returns a set of candidate *target_file* records where the candidate set very likely contains the matching record from a *target_file* if such a record exists. Concord provides an interactive way to specify a blocking function.

4. **Specify a surrogate labeling function** that can automatically label feature function vectors. Surrogate labeling functions are functions that apply a normalized value between 0 and 1.0 to a feature vector in such a way that the high and low values of the surrogate correlate well with ground truth match and mismatch judgements respectively. We call the function a *surrogate* because the output of the function serves as a surrogate for manual judgments. We have found the reciprocal of the block size associated with a given *source_file* and *target_file* record pair is often a good surrogate function. The performance of such functions is clearly conditional upon having identified an effective surrogate.

5. **Select the amount of training data** the Concord system should generate for the machine learner. And **select the type of machine learner** to use. Concord allows one to choose from a variety of machine learning techniques including a support vector machine (SVM) with a linear kernel, an SVM with polynomial kernel, and a logistic regression program.

6. **Instruct the Concord system to use the parameters specified in the previous steps to train a matching model** for resolving *source_file* and *target_file* records and to use the model to perform the actual resolution. The resolution is performed by reading each *source_file* record, retrieving a set of *target_file* records using the blocking function, scoring each feature function vector associated with each *source_file* and *target_file* pair from the block, and writing to an output file the highest scoring record pair in a block.

7. Finally, **review the resolutions** created between *source_file* and *target_file* records and select upper and lower threshold scores. Record pairs associated with a score above the upper threshold are considered matches that require no editorial review. Record pairs with a score below the lower threshold are considered to represent *source_file* records that cannot be matched to a *target_file* because the highest scoring *target_file* record is a mismatch. Record pairs with scores occurring between the upper and lower threshold are considered to be those that require editorial review for accurate categorization into match or mismatch classes.

Figure 3 shows a flow diagram that Concord follows to resolve file records. The flow diagram begins with a source entity record entering the system. Concord extracts from the source record a key (for example, last name of person in record) that it uses to read a block of target records sharing the key. Then Concord takes each target record from the target block and matches it to the source record. The matching is done by computing a feature vector for the source-target record pair and submitting it to the scoring function which computes a match 'belief score.' After all the records in the block have been scored, the highest scoring pair shows the best matching source to target mapping for the particular source record under consideration.

The shaded area of the data flow diagram shows the training cycle. Here the source-target feature vectors are generated just as in the matching cycle but, instead of being scored, feature vectors are labeled with a surrogate label (such as reciprocal of block size). These surrogate labeled vectors are submitted to the machine learner to compute the coefficients of the match model used in the match cycle.

The Concord user interface for resolution system specification is shown in Figure 4.

4.2 Selection of Feature Functions

Resolution systems typically use a set of feature functions to determine how well a *source_file* and a *target_file* record pair match. Feature functions work by comparing a field from the *source_file* record with a field from the *target_file* record and returning a similarity score for the field pair. More formally expressed, a feature function can be represented by the following:

$$F_n(field_{record1}, field_{record2}) = x_n$$

where F_n compares $field_{record1}$ (a field from a *source_file* record) with $field_{record2}$ (a field from a *target_file* record) to compute a value x_n. x_n is an element of the feature vector

$$\vec{x_{i,j}} = x_1, x_2, x_3 ... x_m$$

that is used to compute a comparison score between a *source_file* record i and a *target_file* record j. That is,

$$MatchScore(record_1, record_2) = ScoreVect(\vec{x_{i,j}})$$

Concord allows for the easy selection of feature functions. The developer selects a feature function by specifying the *source_file* and *target_file* fields to compare and then selecting a comparison function from a drop down menu to apply to the fields. Table 1 shows how three feature functions would be specified for matching corporate officer records from two different files. The section of the Concord interface in Figure 4 labeled "Feature Map" is where one can specify feature functions.

Concord provides the developer with a large set of feature functions as well as the ability to add specialized functions one may require for particular resolution problems.

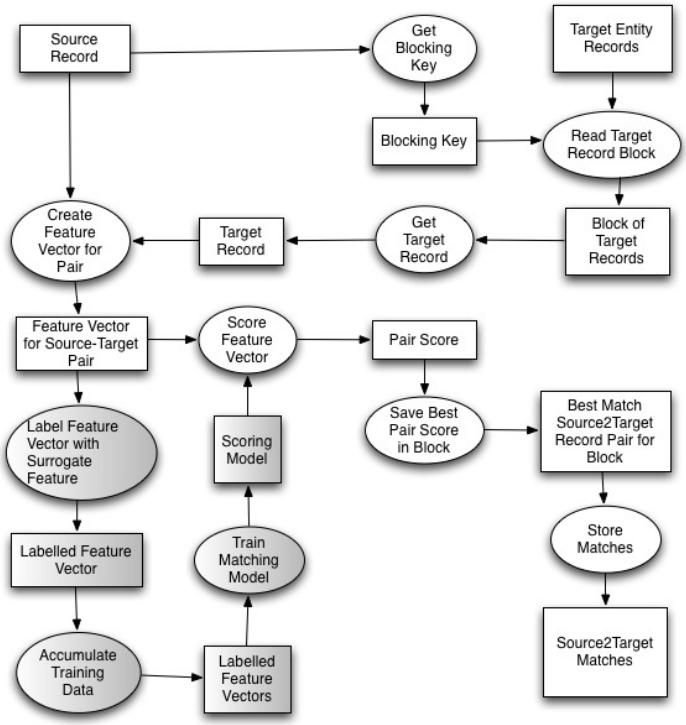

Figure 3: Concord Processing Flow: Training Cycle is Shaded

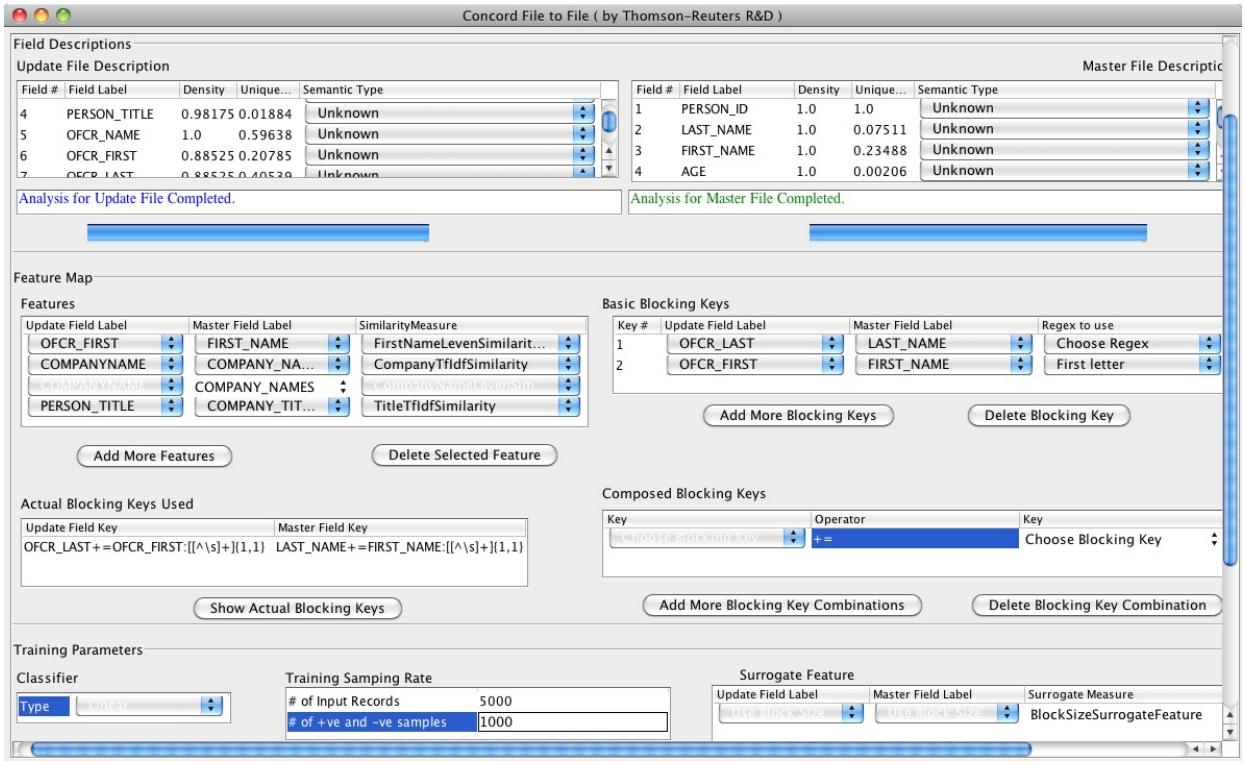

Figure 4: Concord Interface

4.3 Types of Feature Functions

Concord provides five types of feature functions.

1. The first type are string comparison functions that are customized to particular semantic types. These functions include compareFirstNamesLevenshtein(), comparePhoneNumbers(), compareCityState(), compareZipCode(), and compareStreetAddress(). These comparison functions combine string edit distance rules with knowledge of naming conventions associated with particular name types. For example, the compareFirstNamesLevenshtein() feature function gives high match scores to names that have compatible nicknames. Without such a nickname check, many compatible nickname pairs (e.g., William and Bill) would be separated by large edit distances and would thus have low feature match scores.

2. The second type are tf-idf cosine comparison functions whose idf term values are local word level frequencies associated with particular fields in a *source_file* or *target_file*. An example of this is a companyLocalTF-IDF cosine similarity function that uses the frequency counts associated with company name fields in the *source_file* and *target_file* files specified.

3. The third type of feature function is a tf-idf cosine comparison function whose idf term values are taken from word level frequencies associated with large entity name lists outside the framework of a particular resolution problem. An example of this is a companyGlobalTF-IDF cosine similarity function that uses the frequency counts associated with large company name lists that have been assembled from company authority files. So, in this case, for example, we might use a Reuters Company database to compute idf values to use to compare company names in company name fields associated with a pair of relatively small company officer files.

4. The fourth type of function is one from the family of string similarity functions. They include among others the Hamming distance, Levenshtein distance, Smith-Waterman distance, and Jaro-Winkler distance. Each of these functions measures the cost of transforming one string to another as function of the number of character changes that need to be made to transform one string into another. These functions are often robust across different semantic types.

5. The final type of function is a customized function one needs for particular resolution problems. In these cases, Concord allows developers to write their own comparison feature functions and add them to an extensible library of special functions. An example of this type of function might be a function that compares proprietary product codes across two files within a particular company.

By providing developers standard comparison functions for strings and common semantic field types as well as providing the capability of adding specialized functions, Concord strikes a balance between enabling the reuse of standard functions and the customization of functions that may be required for particular resolution systems.

Table 1: Specifications for Three Feature Functions

Source_File Field	*Target_File* Field	Comparison Function
OFCR-FIRST-NAME	FIRST-NAME	FirstNameLevenshtein
COMPANY-NAME	COMPANY-NAMES	CompanyTfIdf-Distance
COMPANY-NAME	COMPANY-NAMES	Levenshtein

4.4 Selection of Blocking Functions

Concord allows for the selection of blocking functions. The purpose of blocking is to increase the efficiency of resolution by limiting the number of *target_file* records to which we must compare a *source_file* record in order to find a match. A good blocking function will return a small mean number of *target_file* records for each *source_file* record under consideration and will also return within the block, with a high probability, the *target_file* record that best matches any given *source_file* record.

A blocking function works by constructing a *source_file* block key from fields in a *source_file* record and reading all *target_file* records that can be indexed by that blocking key value. The blocking key indexes for the *target_file* are built by constructing keys from each *target_file* record using fields in the *target_file* records that are compatible with the fields used to create *source_file* block keys.

An example of a blocking key could be the contents of the last name field in a *source_file* record file of corporate officers and the surname field in a *target_file* record file of corporate officials.

Concord allows developers to specify blocking keys in two steps. First, one specifies a set of basic blocking keys. A basic blocking key shows how a single field or part of a field from *source_file* should be matched against a single field or part of a field from *target_file*. The second part of the blocking specification shows how the basic blocking keys should be assembled to form a final blocking key.

An example of basic and final blocking keys is shown in the interface illustration in Figure 4. Here the basic blocking keys are last name and first initial of the first name. The first basic block key is last name and consists of the field OFCR-LAST from the *source_file* and the field LAST-NAME in the *target_file*. The second basic block key is first initial of first name and consists of first character of field OFCR-FIRST in the *source_file* and first character of field FIRST-NAME in the *target_file*.

In PeopleMap, the blocking key varied by public record type but was often a combination of last name and first digits of zip code.

4.5 Surrogate Function Selection for Training

This subsection addresses a particular application of the surrogate learning technique introduced earlier. The system allows for the selection of a surrogate training function. Surrogate training functions are used to label training data feature functions in lieu of manual judgments (see [9] for more details). Developers can select surrogate functions from the drop down menu labelled Surrogate Feature under the section of the user interface labeled Training Parameters (see bottom of Figure 4).

Surrogate functions must have the characteristic that high values returned by the function correlate on average with true positives (matches) while low values correlate on average with true negatives (mis-matches). Also the surro-

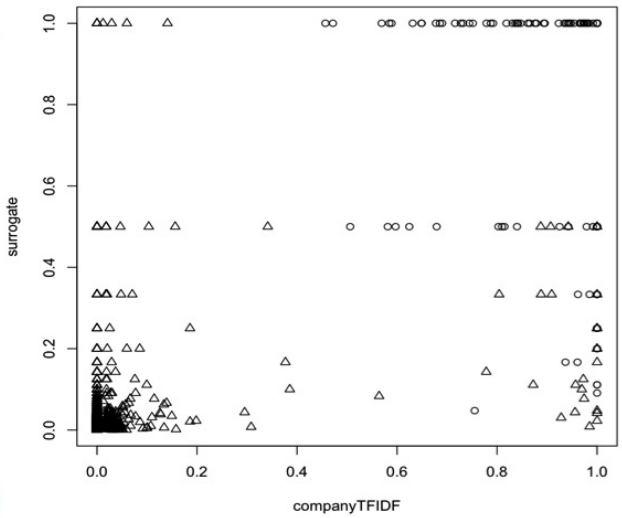

Figure 5: Correlation Between tf-idf and BlockSize

gate function must be class-conditionally independent of the other feature functions.

In our experience, a particularly useful surrogate feature we have found is the reciprocal of the block size, i.e.,

$surrogateFeature = 1/blocksize.$

This works well because in small blocks the ratio of the positive pair to negative pairs is large, the surrogate score is large, one true positive is generated, and a small number of negatives are generated. While for large blocks, the ratio of the positive pair to the negative pairs is small, the surrogate score is small, one true positive is generated, and many negatives are generated. So the mean surrogate score for positive pairs is larger than the mean score for negative pairs. For instance, for a singleton block, the surrogate score would be 1.0 for the single positive feature vector, while, for a block containing 100 records, with 99 true negative feature vectors and 1 true positive, would have a score of 0.01.

The graph in Figure 5 shows the correlation between the reciprocal block size surrogate feature and the company name tf-idf feature for a randomly selected set of record pairs from two different corporate officer files blocked on last name of officer. True positives (matching pairs) are indicated with an o and true negatives (mismatching pairs) are indicated with a $\triangle$. We can see from the distribution of o and $\triangle$ that true negatives are generally associated with both low scoring surrogate features and low scoring company name tf-idf feature values while true positives are much more likely to be associated with high surrogate and high company tf-idf scores.

It turns out that $P(y = 1|x2)$ is a monotonically increasing function of $E[x1|x2]$ for surrogate scores based on the inverse of the block size in many circumstances. Here *x1* is the surrogate inverse block size score and *x2* is the feature vector associated with a record pair. So we can build a regression SVM for $E[x1|x2]$ itself which will rank results in the same order as a classifier that modeled $P(y = 1|x2)$ directly. This relationship is described more fully in [9].

4.6 Training System

Once feature functions, the blocking function, and the surrogate function have been specified, the developer must specify a machine learner and sampling parameters for the generated labelled training feature vectors.

For machine learners, Concord offers one the choice of SVMs with linear, polynomial, and RBS kernels. One can choose his/her machine learners from the drop down menu labelled *Classifiers* under the section of the user interface labeled *Training Parameters*.

Concord samples training feature vectors by randomly choosing a set of n *source_file* records, generating a pool of labeled feature vectors by creating and scoring feature vectors for each *source_file* and *target_file* record pair indicated by the *target_file* record blocks, and then selecting m labeled feature vectors from the generated pool.

The numbers n and m are selected by the developer under the *Training Samples Rate* screen label in Figure 4.

4.7 Record Resolution

The user activates training and resolution of the records in Concord by clicking on a screen button labeled *Train and Resolve*. This causes the Concord system to create a matching model using the training feature vectors and then to resolve each of the *source_file* records to the *target_file*.

Concord performs the resolution of each *source_file* record by reading a block of *target_file* records using the blocking key and scoring each *source_file* and *target_file* record pair in the block using the matching model. The record pair associated with the highest scoring feature vector is stored in the Concord output resolution file. The format of three sample resolution records is shown in Table 2.

A Concord user can browse the resolution results via an output analyzer screen. Figure 6 shows an output analyzer display for the results obtained when we resolved two files containing records describing corporate officers. In this case, the *source_file* contains 860,000 records and a *target_file* contains 840,000 records. The resolution parameters governing this resolution result are shown in Table 3.

Figure 6 shows four ranked result lists. The first list shows *source_file* and *target_file* match pairs ordered by match scores that are above 0.3, the 'high' threshold. The second list shows match pairs with scores between 0.3 and 0.15. The third list shows match pairs with scores below 0.15, the 'low' threshold. The fourth list shows the ids of *source_file* records for which no *target_file* candidate records were found with the blocking key. It is clear in Figure 6 that the upper and lower display thresholds can be set by the user.

The user may scroll through each of these lists and see detailed information about the matched records by clicking on the row list. In Figure 6, the user has clicked on the top scoring record pair in the top list. The screens to the right show detail about the matched pair *source_file* record 778 and *target_file* record 1063283. Both records pertain to a director at Grupo Clarin SA named David Castelblanco [7].

5. EVALUATION

We evaluated PeopleMap with extensive sampling. In each of four separate rounds to testing, either more test records were added or more content types introduced, or both. Our results showed that PeopleMap achieved precision and recall in the mid-90% range, with precision surpassing 95% and recall approaching the same level. These are

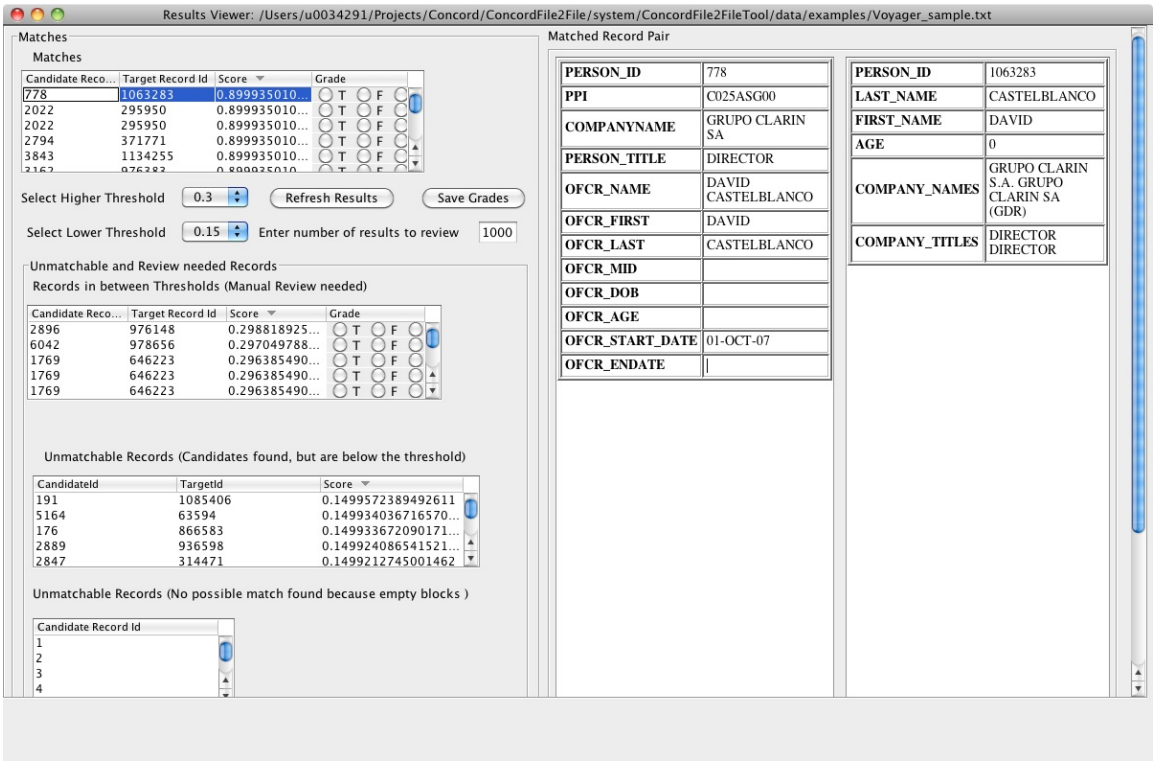

Figure 6: Concord Match Evaluation Interface

Table 2: Format & Example of Resolution Records

Source_File ID	*Target_File* ID	Highest Match Score in Block
123	55301	0.4501
439	67012	0.1534
871	12082	0.8921

arguably near human levels of accuracy. It is worth noting that in any production environment, the resolution engine would allow matches with lower confidence if no matches with high confidence were found. The motivation behind this design was to permit researchers to review additional candidate matches, if they chose to do so.

In addition, in our exemplar corporate officers and directors application, we matched 862,961 *source_file* records to 837,760 *target_file* records. The configuration parameters for the system are shown in Table 3. To assess precision and recall, we tested 300 randomly selected records, and evaluated the highest scoring pairs returned for each block. Table 4 shows precision and recall of the system at various thresholds.

Figure 7 shows how match scores vary by highest scoring record pair for the officers and directors system. A + symbol indicates that the pair is truly a match. A $\triangle$ symbol indicates the pair is a mismatch. Table 4 shows how precision and recall of the officers and directors application vary as the match belief threshold varies. Here a record pair whose score falls above the threshold is deemed a match and any *source_file* record whose best match falls below the threshold is deemed unmatchable to a *target_file* record.

Table 3: Concord Configuration Parameters for Officers and Directors Resolution Experiment

Selection Type	Selected Parameter	Description
File 1	Voyager File	862,962 records
File 2	Reuters File	837,750 records
Feature Function 1	CompareFirstName	edit distance
Feature Function 2	CompareCompanyNameEdit	edit distance
Feature Function 3	CompareCompanyNameTFIDF	tf.idf distance
Feature Function 4	CompareAge	
Feature Function 5	CompareTitle	edit distance
Surrogate Function	Reciprocal of Block Size	
Machine Learner	SVM Linear Regression	

Table 4: Resolution Precision and Recall for Corporate Officers and Directors Problem

Match Score Threshold	Precision	Recall
0.10	0.833	0.966
0.20	0.972	0.852
0.30	0.995	0.808
0.40	0.995	0.800
0.50	1.00	0.792

6. DISCUSSION

As the enabling technology behind PeopleMap, Concord works well for two main reasons. First, many file matching tasks can be solved by applying common feature function field comparison routines. Second, in many cases, the files can be resolved automatically using a technique we call surrogate learning.

The only other system that creates record resolution software we are aware of is the Febrl system [4]. The name

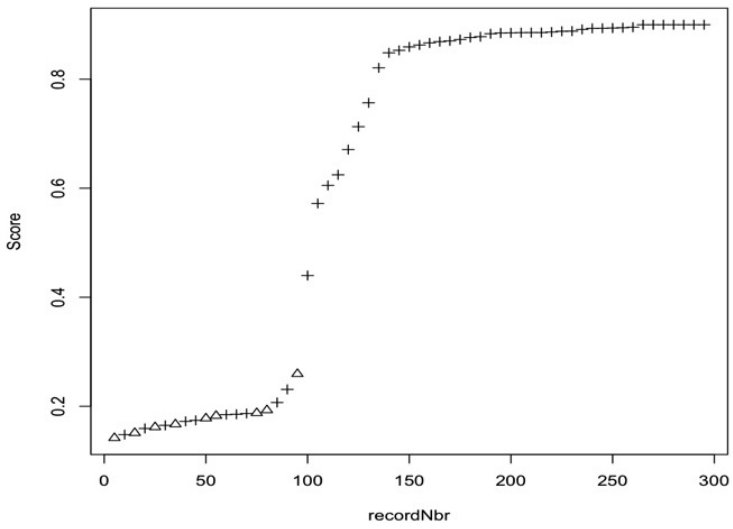

Figure 7: High Match Scores for 300 Officer and Directory Matches. + is true match. △ is mismatch.

Febrl is an acronym standing for Freely Extensible Biomedical Linkage. The Febrl system is written in Python and generates Python record linkage code. The Febrl system is similar to ours in that it provides utilities to analyze the data in file record fields, provides utilities to specify blocking functions, provides a library of feature function routines, and provides a machine learning based method of performing record resolution.[2] It also has utilities to clean and normalize data.

The main differences between Concord and Febrl are the following. Concord provides semi-supervised methods of training a matching function using surrogate labeling. Concord provides a set of feature functions that are customized for particular semantic field types such as person names, street addresses, locations, and company names. Concord feature functions couple field normalization with field comparisons in many cases. And Concord is designed to be able to scale by processing very large files.[3]

The Concord approach and resulting framework has shown itself to be both robust and effective in multiple ways.

1. It has retained the high levels of performance described above as new content sets with a different arrangement of fielded data were added, e.g., when legal docket documents were added to the PeopleMap System;

2. It has proved similarly reliable when deployed in different domains, e.g., in areas of electronic health records;

3. It has demonstrated itself to be robust when relation 'discovery' processes were put in place to exploit second-order and higher relations for treating records that may not have been matched in the original *source_file* processing.

[2]K-means clustering or a SVM classifier can be selected.

[3]The resulting Concord system also contains Java code rather than the Python code associated with Febrl.

7. CONCLUSION

This paper describes PeopleMap, a very large-scale record aggregation system, and Concord, the backbone infrastructure that enables applications like PeopleMap. PeopleMap allows legal professionals to effectively search public records associated with particular people. By contrast, Concord is the behind the scenes software engineering toolkit that allows developers to rapidly create record resolution solutions. The PeopleMap system can link billions of public records to a target authority file of a size comparable to the population of the U.S. It was built using successive applications of Concord to link disparate public record data sets to a central person authority file. It thus represents a powerful and efficient new tool for legal professionals, saving them both time and cost. It permits researchers to identify a single set of search results that formerly required a series of individual searches, one for each content set examined. Furthermore, users would previously be required to manually analyze, merge and prune these disparate search results. With the PeopleMap system, users can produce one integrated person-centric report after as little as a single query. To our knowledge, the PeopleMap system is the largest of its kind.

By contrast, the Concord system is a novel record resolution development tool permitting PeopleMap-like solutions to be developed for other domains.

Some of the key challenges addressed in the research and development of both systems include the following:

- **quality** — harnessing a broad spectrum of complete and partial information across fields in order to maximize precision and recall, with the final PeopleMap application permitting the user ultimately to select the balance between the two;

- **scale** — resolving issues of scale not often confronted before, with O(10B) records being processed and linked to O(100M) authoritative master records;

- **speed** − rapid developing diverse record linkage sys-

tems to establish highly accurate connections between diverse public record sets and a central entity database (e.g., of persons).

In order to be able to devise generalized practical solutions to the challenges posed by problems like PeopleMap, the authors embarked on the development of a more general framework, Concord, which contains a suite of tools that can be leveraged to analyze other disparate data sets, and produce highly accurate record-linkages as efficiently as possible.

8. FUTURE WORK

There are several different directions future deployments of the Concord system can take, as can the underlying research that fuels its effectiveness and utility. The most obvious of these includes the harnessing of new document-types and the resulting public records contained within (e.g., watch lists for fraud or terrorism, e-commerce records, compliance lists). Other directions include applications in new domains for other distinct business units. These domains could include science (e.g., scientists, their fields of study, their institutions) or medicine (doctors, areas of specialty, their health care organizations). Other directions include internationalization (e.g., using authority files that are non-U.S.-based) and the deployment of Concord using foreign language authority files and records (e.g., in Spanish, Russian, Korean). And still others include using higher order relations to discover other associations between records or profiles and the people represented in our authority files.

9. ACKNOWLEDGMENTS

The authors are grateful to Peter Jackson and Khalid Al-Kofahi for their sustained support of this research. They also thank Kevin Appold, Paul Wohletz and Michelle Cabbage for providing on-going domain expertise for the PeopleMap project. They acknowledge the early contributions of Steve Samler and the on-going contributions of Gary Berosik and Arun Vachher. In addition, the authors are appreciative of the system assessments provided by Greg Becker and James Votel.

10. REFERENCES

[1] R. Baxter, P. Christen, and T. Churches. A comparison of fast blocking methods for record linkage. In *Proceedings of the First Workshop on Data Cleaning, Record Linkage and Object Consolidation*, page 6 pgs., Washington, D.C., August 2003. KDD.

[2] O. Benjelloun, H. Garcia-Molina, D. Menestrina, Q. Su, S. E. Whang, and J. Widom. Swoosh: A generic approach to entity resolution. *International Journal on Very Large Data Bases (VLDB J.)*, 18(1):255–276, 2009.

[3] I. Bhattacharya and L. Getoor. Iterative record linkage for cleaning and integration. In *Proceedings of the 9th ACM SIGMOD Workshop on Research Issues in Data Mining and Knowledge Discovery (DMKD 2004)*, pages 11–18. ACM Press, 2004.

[4] P. Christen. Febrl - a freely available record linkage system with a graphical user interface. In *Second Australasian Workshop on Health Data and Knowledge Management*, Wollongong, NSW, Australia, 2008.

[5] W. W. Cohen, P. Ravikumar, and S. E. Fienberg. A comparison of string distance metrics for name-matching tasks. In *Proceedings of IJCAI-03 Workshop on Information Integration*, pages 73–78. American Association of Artificial Intelligence (AAAI), August 2003.

[6] W. W. Cohen and J. Richman. Learning to match and cluster large high-dimensional data sets for data integration. In *Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '02)*, pages 475–480, New York, NY, USA, July 2002. ACM.

[7] C. Dozier and R. Haschart. Automatic extraction and linkage of person names in legal text. In *Proceedings of RIAO 2000 (Recherche d'Information Assistée par Ordinateur)*, pages 1305–1321, Paris, France, April 2000. RIAO.

[8] A. McCallum, K. Nigam, and L. H. Ungar. Efficient clustering of high-dimensional data sets with application to reference matching. In *Proceedings of the Sixth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '00)*, pages 169–178, New York, NY, USA, 2000. ACM.

[9] S. Veeramachaneni and R. Kondadadi. From feature independence to semi-supervised classification. In *Proceedings of the NAACL Workshop on Semi-Supervised Learning*. North American Association of Computational Linguistics, 2009.

[10] V. S. Verykios, G. V. Moustakides, and M. G. Elfeky. A bayesian decision model for cost optimal record matching. *International Journal on Very Large Data Bases (VLDB J.)*, 12(1):28–40, 2003.

[11] W. E. Winkler. Advanced methods for record linkage. In *Proceedings of the Section on Survey Research Methods*, pages 467–472. American Statistical Association, 1994.

[12] W. E. Winkler. The state of record linkage and current research problems. In *Proceedings of the Survey Methods Section*, pages 73–80. Statistical Society of Canada, 1999. Longer version available at http://www.census.gov/srd/www/byyear.html.