RedFame

Applied Economics and Finance
Vol. 3, No. 1; February 2016
ISSN 2332-7294 E-ISSN 2332-7308
Published by Redfame Publishing
URL: http://aef.redfame.com

An Algorithm for Matching Heterogeneous Financial Databases: A Case Study for COMPUSTAT/CRSP and I/B/E/S Databases

Irene Rodriguez-Lujan[1] & Ramon Huerta[2]

[1]Machine Learning Group, Escuela Politécnica Superior, Universidad Autónoma de Madrid, 28049 Madrid, Spain

[2]Rady School of Management, University of California, San Diego, La Jolla, CA 92093

Correspondence: Ramon Huerta, Rady School of Management, University of California, San Diego, La Jolla, CA 92093.

Received: October 26, 2015 Accepted: November 12, 2015 Available online: February 1, 2016

doi:10.11114/aef.v3i1.1164 URL: http://dx.doi.org/10.11114/aef.v3i1.1164

Abstract

Rigorous and proper linking of financial databases is a necessary step to test trading strategies incorporating multimodal sources of information. This paper proposes a machine learning solution to match companies in heterogeneous financial databases. Our method, named Financial Attribute Selection Distance (FASD), has two stages, each of them corresponding to one of the two interrelated tasks commonly involved in heterogeneous database matching problems: schema matching and entity matching. FASD's schema matching procedure is based on the Kullback-Leibler divergence of string and numeric attributes. FASD's entity matching solution relies on learning a company distance flexible enough to deal with the numeric and string attribute links found by the schema matching algorithm, and it incorporates different string matching approaches such as edit-based and token-based metrics. The parameters of the distance are optimized using the F-score as cost function. FASD is able to match the joint Compustat/CRSP and Institutional Brokers' Estimate System (I/B/E/S) databases with an F-score over 0.94 using only a hundred of manually labeled company links.

Keywords: Compustat/CRSP, I/B/E/S, financial data, heterogeneous databases, company matching, schema matching, attribute matching, Kullback-Leibler divergence

1. Introduction

The increasing volume of data during last year's arises new challenges for machine learning and databases communities focused on the efficient integration, interoperability, and comparative of data originated from different sources and stored in heterogeneous databases. Financial data analysis is the domain where these problems are especially challenging given the evident relevance of the data, the ubiquitous presence of sources of information, and the inherent difficulty of dealing with noisy, spurious or missing data. The existence of different agencies or institutions independently reporting their financial data together with the use of their own concepts, vocabulary, and company identifiers make the manual integration of these databases time consuming and dependent on expert knowledge. Therefore, the implementation of machine learning algorithms capable of automatically matching financial databases is almost mandatory. In this paper we will match the joint Compustat/CRSP database and the Institutional Brokers' Estimate System (I/B/E/S) History database. The joint Compustat/CRSP database contains information about the fundamentals of a company, while the I/B/E/S History database provides both detailed and summarized analysts' estimates of up to 26 forecast measures (earnings per share, revenue/sales, net income, among others) as well as information about the fundamental data reported by the companies. Each company in these databases is defined by a set of time-stamped records formed by a set of string and numeric attributes such as company name and fundamental data. However, the direct identification of joint Compustat/CRSP and I/B/E/S variables according to their definitions is not straightforward and more sophisticated approaches are required. In addition, each database has its own primary identifiers associated with each company or security, which makes databases merging more complicated. Therefore, two interrelated tasks need to be solved (Liu, Dou, & Wang, 2012; Zhao & Ram, 2007): a *schema matching* (*attribute matching*) problem responsible for finding correspondences between attributes, and an *entity matching* (*object matching* or *record linkage*) problem oriented to discovering links between companies based on the relationships found by the schema matching procedure. However, the entity matching problem associated with financial databases is different from entity matching problems commonly addressed in the literature. The entity matching problem is usually defined as the problem of determining whether two entities in different databases refer to the same object. In other words, entity

matching algorithms try to discover one-to-one relationships between records since they assume that an object is represented by a single record. However, in financial databases the objects are companies with multiple records in the database. Typically, each of these records in the database has a time stamp that indicates the period of time to which the data is referred.

The main goal of this work is to provide a general machine learning framework to link companies in heterogeneous financial databases. The databases are linked by maximizing the F-score to aid in the construction of predictive modeling tools and other applications (Huerta, Elkan, & Corbacho, 2013; Kim & Han, 2000; Sewell, 2010). The FASD system consists of two algorithms, each of them responsible for solving the attribute matching and company matching problems. The goal of our schema matching algorithm is to provide an automatic procedure to obtain one-to-one links between attributes in Compustat/CRSP and I/B/E/S databases without using any kind of prior knowledge to reduce as much as possible the human supervision. FASD uses the Kullback-Leibler divergence (KL) as similarity measure between fields in different databases. This is an unsupervised algorithm in which we only consider the distribution of the values of the attributes without using neither temporal nor entity matching information. Additionally, this approach takes advantage of the large amount of data available in financial databases and relies on the assumption that matching attributes should have similar probability distributions given that we expect a high overlapping between both databases. Among different approaches oriented to solve the attribute matching problem (Bernstein, Madhavan, & Rahm, 2011; Gal & Shvaiko, 2009; Shvaiko, 2005; Rahm & Bernstein, 2001; Doan, Domingos, & Halevy, 2001), Kullback-Leibler divergence has been previously used to find links between continuous numeric and discrete-valued attributes (Jaiswal, Miller, & Mitra, 2013; Jaiswal, Miller, & Mitra, 2010) and mutual information has also been suggested to find attribute links (Kang & Naughton, 2008; Zhao, 2010). However, FASD's schema matching solution is the first attempt to use Kullback-Leibler divergence to link string and numeric attributes in heterogeneous databases. To solve the company matching problem, FASD finds the optimal parameters of a generalized company matching measure capable of (i) taking into account the temporal information inherent to financial data, (ii) determining which of the links obtained by the schema matching procedure are really useful, (iii) choosing the best distance for each string attribute among three well-known string matching measures, (iv) combining string matching and token-based distances, (v) considering word order and possible bad ordering of the sequence of words, and (vi) establishing the optimal threshold that determines whether two companies are the same according to our company-similarity metric. Supervised learning of the generalized distance is especially suitable for reducing human effort and expert knowledge since our similarity function admits many different configurations that do not have to be manually tuned. Unlike our schema matching approach, the company matching algorithm is supervised, so it needs as inputs some pairs of linked and/or unlinked companies. Though there exist numerous approaches in the literature to solve the entity matching problem (Köpcke & Rahm, 2010; Köpcke, Thor, & Rahm, 2010; Elmagarmid, Ipeirotis, & Verykios, 2007; de Carvalho, Laender, Goncalves, & da Silva, 2012; Isele & Bizer, 2012; Camacho, Huerta, & Elkan, 2008), they do not provide the FASD's flexibility to handle heterogeneous data types and temporal data, and they generally need to be manually tuned and configured (Köpcke & Rahm, 2010; Köpcke, Thor, & Rahm, 2010; Köpcke & Rahm, 2008).

The rest of the paper is organized as follows. Section 0 describes our schema and entity matching algorithms. Section 0 presents the Compustat/CRSP and I/B/E/S data used in our experiments, it describes the experimental setup, and it shows the results obtained by the FASD algorithm. Finally, Section 0 states the conclusions derived from this work.

2. Method

As discussed above, the Financial Attribute Selection Distance algorithm (FASD) has two stages, each of them corresponding to one of the two interrelated tasks commonly involved in heterogeneous database matching problems: schema matching and entity matching.

2.1 Schema Matching

Given two databases, our schema matching algorithm gives as an output a ranking of the most likely links between pairs of attributes according to their Kullback-Leibler divergence. Financial databases can be formed by both numerical and string-type fields, but only matches between fields of the same type are considered. Therefore, according to the standard taxonomy to classify schema matching techniques (Bernstein, Madhavan, & Rahm, 2011; Rahm & Bernstein, 2001), FASD's matching strategy can be categorized as a instance-matching approach since it determines the similarity of attributes according to a statistical measure, but it also can be considered as a constraint-based approach as only matchings between attributes of the same type (e.g. string or numeric) are analyzed. The instance-matching approach is especially suitable for these financial databases in order to take advantage of the large amount of information available. We adopt the discrete Kullback-Leibler divergence even for continuous numerical features because of its easy implementation. For string-type attributes, the distribution of each attribute is obtained by considering the distribution of the words appearing in either of the two attributes to compare. For numeric-type attributes, the continuous probability distribution is approximated by a histogram with N evenly spaced bins, where N is a meta-parameter of the schema

matching algorithm. In all cases, only non-missing data were used to obtain the histograms. Specifically, given two attributes A_i^1 and A_j^2 from two different databases, and their discrete probability distributions $\{p_i\}_{i=1}^N$ and $\{q_i\}_{i=1}^N$, respectively, the Kullback-Leibler divergence can be defined as follows:

$$D_{KL}(A_i^1, A_j^2) = \sum_{k=1}^{N} p_k \log \frac{p_k}{q_k} \ . \quad (1)$$

The KL divergence is an asymmetric measure, and we use its symmetric version expressed as follows:

$$D_{KL}^{sym}(A_i^1, A_j^2) = \frac{1}{2}\left(D_{KL}(A_i^1, A_j^2) + D_{KL}(A_j^2, A_i^1)\right) \ . \quad (2)$$

Once the Kullback-Leibler divergences are computed for all the possible pairs of attributes of the same type (string/numeric), we apply a filter in order to avoid spurious links. In particular, we define two thresholds for the entropy of the attributes (ε_{ent}) and the percentage of missing values (ε_{mv}), respectively. We are not interested in matching attributes with low entropies or high percentage of missing values since they do not contain enough information to reliably estimate probability distributions. The output of FASD's first stage is a ranking of filtered attribute links sorted according to their KL divergence; the lower KL divergence, the better. FASD's schema matching algorithm is fully-automatic and does not require any kind of human effort except for selecting the final number of links. The user can keep as many links as she/he wants from the ranking of attribute links since the entity matching algorithm described in the following section will determine which of these links are indeed useful for company matching purposes.

2.2 Company Matching

The main advantage of FASD's company matching approach is its versatility. Companies are matched as a function of a self-tuning company distance whose optimal parameters are adjusted by optimizing a cost function (metaheuristic). The cost function that we use to guide the optimization process is the F-score (or F-measure). The larger F-score, the better; thus, the FASD's objective is to maximize the F-score. The F-score is defined as the weighted harmonic mean of *precision* and *recall* measures commonly used in Information Retrieval (Baeza-Yates, Ribeiro-Neto, & others, 1999). In our problem, precision is the fraction of correspondences identified by our algorithm that are indeed correct, while recall measures the fraction of correct correspondences identified by our algorithm over all the possible correspondences. That is,

$$F\ score = \frac{2 \times Precision \times Recall}{Precision + Recall} \ , \quad (3)$$

$$Precision = \frac{TP}{TP + FP} \ , \quad (4)$$

$$Recall = \frac{TP}{TP + FN} \ , \quad (5)$$

where TP (True Positives) is the number of links correctly identified, FP (False Positives) is the number of links incorrectly identified, and FN (False Negatives) is the number of positive links not identified. F-score ranges between 0 and 1, where 0 and 1 correspond to total failure and perfect retrieval scenarios, respectively. Parameters maximizing the F-score are determined by a Genetic Algorithm (Holland, 1975) for convenience, but any other metaheuristic optimization techniques such as simulated annealing (Duda, Hart, & Stork, 2012) may be applied. In the following subsections, we will build FASD's company distance, which can be defined by four levels in a bottom-up architecture:

- **Level 1** (bottom) is responsible of computing the distance between two records with the same time stamp and only considering one attribute or field.
- **Level 2** combines distances from Level 1 by assigning different weights to databases' fields.
- **Level 3** averages distances from Level 2 over the set of intersecting time stamps.

- **Level 4** finds the optimal threshold to determine whether or not two companies are linked.

2.2.1 Level 1: String Distance and Numeric Distance Functions

We implemented a general string distance that is the convex combination of an edit distance metric and a token-based (or statistical) metric. On the one hand, edit distance metrics (also known as string matching methods) quantify differences between strings as a function of the number of character insertions and deletions needed to transform one string into the other one. Examples of these metrics include the Levenshtein (Levenshtein, 1966), Jaro (Jaro, 1989), Jaro-Winkler (Winkler, 1999), and Monge-Elkan (Monge & Elkan, 1997; Monge, Elkan, & others, 1996) approaches. The success of these methods relies on the assumption that same concepts are likely to be modeled with similar names. This assumption is useful in financial matching problems as it is expected that, for example, the same company has similar names in both databases. On the other hand, statistical metrics (also known as token-based distances) are based on an underlying statistical model that somehow measure the importance of a word in a document. Jaccard similarity, cosine distance (or TF-IDF), and Soft TF-IDF are some examples of token-based metrics (Bilenko, Mooney, Cohen, Ravikumar, & Fienberg, 2003; Cohen, Ravikumar, & Fienberg, 2003). The integration of statistical distances is also beneficial since there are several terms commonly present in company names that are not so discriminative for matching purposes; for example, terms such as ``corporation" and ``incorporated". In addition, our string distance also takes into account the order of words in the string by assuming that first words are more relevant, and it also integrates the Monge-Elkan concept to consider possible mismatches due to bad ordering of the sequence of words.

In what follows, we assume that databases are only formed by attributes selected according to the schema matching algorithm, and these attributes (or fields) are arranged so that the first field in both databases corresponds to the first attribute link and so on. Let $F_i^m(t)$ and $F_j^m(t)$ two strings corresponding to the m-th field of the company identified as i in the first database and the m-th field of a company identified as j in the second database. We assume that both records have the same time stamp t. $F_i^m(t)$ and $F_j^m(t)$ can be split in two set of words $\{w_i^k(t)\}$ for $k = 1, 2, \dots, N_i$ and $\{w_j^k(t)\}$ for $k = 1, 2, \dots, N_j$, respectively. Let $N = \max(N_i, N_j)$, we complete the shortest string with empty words up to have N words. In order to accelerate the learning algorithm, we implemented the Monge-Elkan concept by applying the permutation operation P_l with the restriction that we only permute the first three words of each string. Then, the string distance between $F_i^m(t)$ and $F_j^m(t)$ can be written as follows

$$\mu_{str}\left(F_i^m(t), F_j^m(t)\right) = \min_l \left\{ \sum_{k=1}^{N} \frac{e^{-\gamma_m \cdot k}}{\sum_{k'=1}^{N} e^{-\gamma_m \cdot k'}} \left[\alpha_m \cdot \mu_{edit}^m \left(w_i^k(t), w_j^{P_l(k)}(t) \right) + (1 - \alpha_m) \cdot \mu_{token}^m \left(w_i^k(t), w_j^{P_l(k)}(t), S_m \right) \right] \right\}. \quad (6)$$

The term S_m refers to the corpus formed by all the words in all the entries of the m-th field in both databases. Parameter α_m is a positive value that regulates the convex combination of the edit distance (μ_{edit}) and the token-based distance (μ_{token}), while parameter γ_m is also a positive real number that adjusts the weight (importance) of each word assuming that first words are more relevant. Our edit distance allows the optimization algorithm to choose the best among three well-known edit distances namely, Levenshtein (Levenshtein, 1966), Jaro (Jaro, 1989), and Jaro-Winkler (Winkler, 1999), and it can be expressed as follows

$$\mu_{edit}^m \left(w_i^k(t), w_j^{P_l(k)}(t) \right) = d_{type}^m \left(w_i^k(t), w_j^{P_l(k)}(t) \right), \quad (7)$$

where d_{type}^m is a parameter that takes three discrete values to represent the three edit-distance metrics (Levenshtein, Jaro, and Jaro-Winkler). Nevertheless, it is straightforward to incorporate any other matching string distances. On the other hand, our token-based distance accounts for the context statistics by using the *Inverse Document Frequency* score (IDF), which determines how common a word is across all the records. We normalize the IDF score to have it in the interval $[0,1]$, making both terms in the convex combination in Equation (6) similar in magnitude. Our token-based distance can be expressed as follows

$$\mu_{token}^m \left(w_i^k(t), w_j^{P_l(k)}(t), S_m \right) = IDF_{norm} \left(w_i^k(t), S_m \right) \cdot IDF_{norm} \left(w_j^{P_l(k)}(t), S_m \right), \quad (8)$$

$$IDF_{norm}(w,S) = \frac{IDF(w,S)}{\sqrt{\sum_{w' \in S} IDF(w',S)}}\,, \tag{9}$$

$$IDF(w,S) = \log \frac{\#\ records\ in\ the\ corpus\ S}{\#\ records\ in\ S\ containing\ the\ word\ w + 1}\,. \tag{10}$$

Finally, the numeric distance between two numeric fields $F_i^m(t)$ and $F_j^m(t)$ with the same time stamp t is given by the exponential of the relative error between both fields. The exponential penalization guarantees that only those companies whose fundamental data are very similar are considered as possible matchings. The numeric distance is defined as follows

$$\mu_{num}^m\left(F_i^m(t), F_j^m(t)\right) = 1 - \exp\left(-\left\|\frac{F_i^m(t) - F_j^m(t)}{F_i^m(t)}\right\|\right). \tag{11}$$

Note that the numeric distance can be further optimized by weighting the exponential term by a factor. However, this would require another metaparameter to search and the current solution gives good results as shown below.

2.2.2 Level 2: Record Distance

Once we have defined the string and numerical distances for each individual field, we combine these magnitudes to define the distance between two records with the same time stamp t by adjusting a parameter $\eta_m \in [0,1]$ for each field. These parameters represent the weight (or importance) of each field in the global score. Therefore, our record distance between two entries $F_i(t)$ and $F_j(t)$ with the same time stamp t and represented by M fields is given by the weighted average over all the fields. It can be written as follows

$$\begin{aligned}\mu_{rec}\left(F_i(t), F_j(t)\right) = \frac{1}{\sum_{m'=1}^{M} \eta_{m'}} \sum_{m=1}^{M} \eta_m (isString(m)) \cdot \mu_{str}\left(F_i(t), F_j(t)\right) + (1 - isString(m)) \\ \cdot \mu_{num}\left(F_i^m(t), F_j^m(t)\right),\end{aligned} \tag{12}$$

where $isString(m)$ is a predicate that takes the value 1 when the m-th field in both databases is string-type, and 0 otherwise. It is worth noting that both the string and numeric distances take values in the interval $[0,1]$, and, thus, all the terms in (12) have the same range of values.

2.2.3 Level 3: Company Distance

We define the distance between two companies as a function of the distance between their records. Given that one company is defined by a set of records with an associated time stamp, the company distance is defined as the average record distance (Equation (12)) between those pairs of records sharing the same time stamp. Let $F_i = \{F_i(t)\}$ for $t \in T_i$ be the records corresponding to the company i in the first database, and let $F_j = \{F_j(t)\}$ for $t \in T_j$ be the entries associated with the company j in the second database. The FASD's company distance is given by the following expression

$$\mu_{comp}(F_i, F_j) = \frac{1}{\left|\{T_i \cap T_j\}\right|} \sum_{t \in \{T_i \cap T_j\}} \mu_{rec}\left(F_i(t), F_j(t)\right), \tag{13}$$

where $\{T_i \cap T_j\}$ is the set of time stamps with records in both databases and $|\cdot|$ denotes the cardinality of this set.

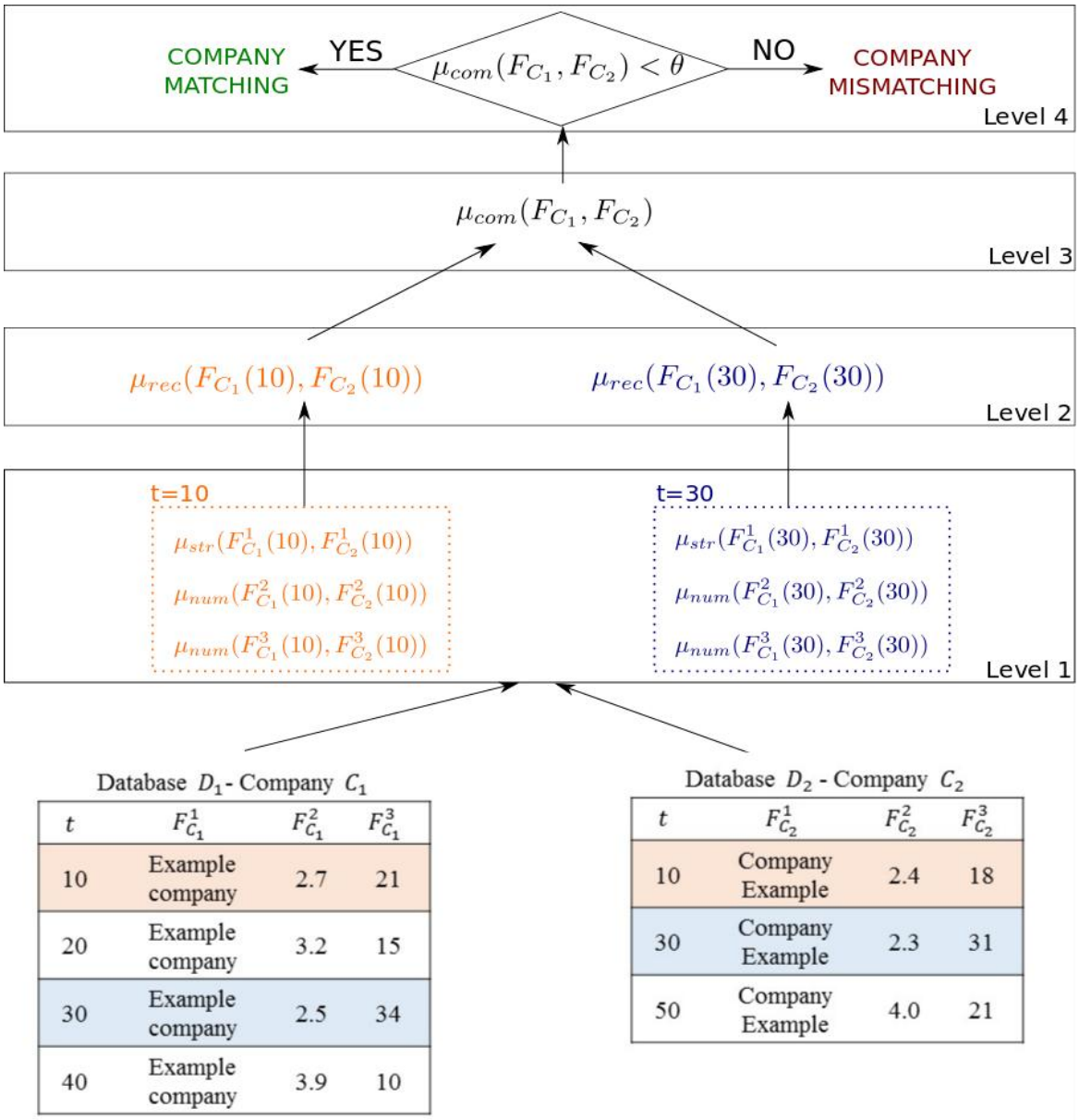
2.2.4 Level 4: Matching Rule

According to the company distance defined by (14), we classify two companies as a match when their company distance is below a threshold θ, which is automatically tuned; that is, two companies F_i and F_j are identified as the same company if $\mu_{comp}(F_i, F_j) < \theta$. After putting all the levels together, we end up with five parameters to be

optimized. For the sake of clarity, we summarize them in Table 1.

Table 1. FASD distance parameters

Distance	Parameter	Description	Range
μ_{str}	γ_m	Weight of each word in the m-th field according to its position in the string	[0,7]
μ_{str}	α_m	Parameter of the convex combination between edit and token based distances for the m-th field	[0,1]
μ_{edit}^{m}	d_{type}^{m}	Type of edit distance metric to be used in the m-th field	{0,1,2}
μ_{rec}	η_m	Weight of the m-th field	[0,1]
μ_{comp}	θ	Threshold of the company matching function	[0,1]



Database D_1 - Company C_1

t	$F_{C_1}^1$	$F_{C_1}^2$	$F_{C_1}^3$
10	Example company	2.7	21
20	Example company	3.2	15
30	Example company	2.5	34
40	Example company	3.9	10

Database D_2 - Company C_2

t	$F_{C_2}^1$	$F_{C_2}^2$	$F_{C_2}^3$
10	Company Example	2.4	18
30	Company Example	2.3	31
50	Company Example	4.0	21

Figure 1 Example of the four levels of the FASD's company matching approach

In order to provide a general overview of our company matching approach, Figure 1 shows a simplified example of the four levels of the company matching algorithm given the three attribute correspondences F^1, F^2 and F^3, and the entries for companies C_1 and C_2 in databases D_1 and D_2, respectively.

3. Results

The aim of this section is to describe the data used in the experiments and present the results obtained in the evaluation of FASD's schema and company matching algorithms proposed in Sections 0 and 0, respectively. The C++ code for the schema and company matching algorithms are available upon request. The PGA-PACK library (Levine, 1996) was used to implement the optimization. Though there exist some frameworks to automatically constructing entity matching strategies (Köpcke & Rahm, 2008), the results shown below cannot be compared to other methods because of the lack of a generic framework to compare heterogeneous databases with the characteristics of the financial domain: time-stamps at different scales, text, numbers, and non-intersecting fields at different degrees.

3.1 Results on Schema Matching

To identify the relationship between the Compustat/CRSP and I/B/E/S attributes, *Actuals Data* from I/B/E/S Summary History was employed. According to I/B/E/S Summary History User Guide - April 2013 version 3, "Actuals Data provides financial data relating to previous fiscal periods as reported by the company. Actuals for fiscal periods and interim periods are obtained from news services and company filings and adjusted by Thomson Reuters market specialists to be comparable to the estimates made by analysts". All the available quarterly data from 2008 to 2012 were used. Information prior to 2008 was not included since some I/B/E/S variables were not available until 2008. All quarterly data features (332 attributes) and all year-to-date features (232 attributes) as well as 'Company Name' (CONAME) and 'Company Legal Name' (COLEGNAME) fields from Compustat/CRSP database form the inputs to the algorithm. All the non-industry specific measures available in the I/B/E/S Summary History were included together with the 'Long Company Name' field extracted from Company Identification files.

Table 2. Description of Compustat/CRSP and I/B/E/S data used in FASD

Database	#Records	#Keys (companies)	#String	#Numeric
Compustat/CRSP	111 630	7216	2	564
I/B/E/S	91 694	7561	1	24

Table 2 shows the total number of records, company identifiers (keys), string attributes, and numeric attributes provided to FASD. In the Compustat/CRSP database, we used the field Fiscal Year (FYEARQ) to keep only data referred to the period 2008-2012, and we used as time stamp for each record the Data Date (DATADATE) field. For the I/B/E/S database, we considered the 'Period End Date' field as time stamp.

Table 3. Kullback-Leibler divergence of the best four disjoint numeric-type and the best string-type links between Compustat/CRSP and I/B/E/S databases

Link type	Compustat/CRSP attribute	I/B/E/S attribute	Kullback-Leibler Divergence
Numeric	Revenue Total (revtq)	Revenue, non-per-share (SAL)	$3.56 \cdot 10^{-3}$
Numeric	Net Income (Loss) (niq)	Reported Net Profit (NER)	$3.84 \cdot 10^{-3}$
Numeric	Pretax Income (piq)	Reported Pretax Profit (PRR)	$8.44 \cdot 10^{-3}$
Numeric	Pretax Income after depreciation (oiadpq)	Operating Profit, non-per-share (OPR)	$8.87 \cdot 10^{-3}$
String	Company legal name (COLEGNAME)	Long company name (CONAME)	2.51

The best four disjoint numeric-type links corresponding to four different attributes in Compustat/CRSP database and four different attributes in I/B/E/S database together with the best string-type matching are shown in Table 3. We set the number of bins N equals to 100, the entropy threshold ε_{ent} equals to 10^{-2}, and the missing value threshold ε_{mv} equals to 50%. Before invoking FASD, in order to homogenize company name, we applied a dictionary based expansion to extend the most common acronyms; for instance, "inc" was replaced by "incorporated", and "ltd" was substituted by "limited".

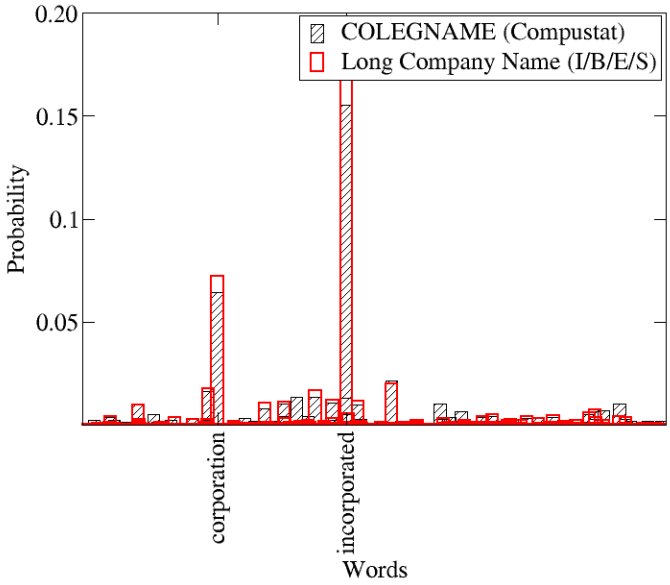


Figure 2. Distribution of words for attributes 'COLEGNAME' and 'CONAME' in Compustat/CRSP and I/B/E/S databases, respectively. Each bin represents one word. There are 7483 bins in total.

As an example, Figure 2 shows the histogram of the attributes 'COLEGNAME' and 'CONAME' in Compustat/CRSP and I/B/E/S databases, respectively. Common terms such as "incorporated" or "corporation" have probabilities of occurrence of 0.17 and 0.16 in Compustat/CRSP and I/B/E/S databases, respectively, which diminishes its discriminative power and motivates the use of token-based distances.

3.2 Results on Company Matching

The best four numeric-type links and the best string-type link presented in Table 3 were used as input to FASD's company matching algorithm described in Section 0. Taking into account these heterogeneous attributes is fundamental to obtain a good performance, as it will be shown further down. Though there are many *consistent* cases in which company names and fundamental data are very similar, there also exist other scenarios in which either the company names are very similar but there are discrepancies in the numerical attributes, or company names are significantly different but fundamental data are almost identical. We filtered the data shown in Table 2 by dismissing those records containing a missing value in any of the five attributes chosen. The data used is shown in Table 4.

Table 4. Description of the Compustat/CRSP and I/B/E/S information used for company matching. The average (Avg.), minimum (Min.), and maximum (Max.) number of records per company are also provided.

Database	#Records	#Keys (companies)	#String	#Numeric	Avg.	Min.	Max.
Compustat/CRSP	85 476	5695	1	4	15.01 ± 6.60	1	21
I/B/E/S	41 212	4447	1	4	9.27 ± 5.35	1	24

Since FASD's company matching algorithm is a supervised algorithm, we need to provide it with a subset of positive and negative company links. We can obtain a partial list of company matchings by using the historic CUSIP number (Commission, s.f.) available in both databases by following the procedure provided by Wharton Research Data Services (WRDS) (Moussawi, 2006). Historical CUSIP numbers allow recovering information about the companies. The first six digits uniquely identify the company, while the last two digits represent the company's stock issue. Moreover, CUSIP numbers are never reassigned to other companies or stocks. We linked companies using the most recent CUSIP in order to have a list of company matchings that allow us to evaluate FASD's effectiveness. However, CUSIP numbers are not always available or completely correct, but they are very useful for calibrating the parameters of FASD.

We want to evaluate the performance of FASD's company matching algorithm in terms of its effectiveness to identify company links and the human labeling cost. Therefore, our results will present the F-score of FASD's company matching approach when different number of company links labeled as linked or unlinked are used in training. At this point, it is worth noting that there may exist a few N-to-N links between companies, and we do not expect a perfect matching since there are some companies present in Compustat/CRSP that are not in I/B/E/S, and vice versa. It is known that the effectiveness of a supervised entity matching methods highly depends on the size and quality of the

training data, which ideally should reproduce the real operating scenario (Köpcke & Rahm, 2008; Köpcke, Thor, & Rahm, 2010). We applied three different strategies to select n training pairs of positive and negative company links to be used by FASD:

- **Random selection**. As a baseline error, we randomly choose $n/2$ positive pairs from the list of known links, and we randomly choose $n/2$ unlinked pairs. Please note that this strategy does not represent a feasible scenario in practice since linked and unlinked companies are not known beforehand. However, a *pure random selection* would be entirely dominated by negative links. For example, in our data (Table 4), we can have at most 4447 positive pairs assuming one-to-one correspondences, while the number of negative links is $5695 \times 4447 - 4447 = 25\,321\,218$.
- **Deterministic selection**. We take the n pairs with the lowest distance, computed according to (15) and using uninformative parameters ($\gamma_m = 0$, $\alpha_m = 0.5$, $d^m_{type} = 3$ (Jaro-Winkler), and $\eta_m = 1/M$), to be manually labeled. A value for the threshold θ is not needed to compute the distance.
- **Heuristic selection**. As an upper bound, we consider that we have at most L links between companies with L the maximum number of companies per database. In our case, $L = \max\{5695, 4447\} = 5695$. We randomly select n candidate pairs to be manually labeled among the $2L$ pairs with the lowest company distance with uninformative parameters.

The performance of the algorithm is evaluated over a test set formed by all possible company links obtained by the cross product between companies. Though the complete list of positive links is not available, we used as ground truth the positive links obtained by the CUSIP number procedure described above and the positive links found by manually labeling the first $2L$ pairs with the lowest uninformative company distance. Three different sets of attributes are considered: (i) the best four numeric-type links and the best string-type link, (ii) the best four numeric links, and (iii) the best string link. The total number of positive links with data in both databases is 3932.

Table 5. F-score of FASD's company matching algorithm as a function of the number of training samples n and using different training set selection strategies. The best four links for numeric attributes and the best link for string attributes found by FASD's schema matching algorithm are considered. Results for the random and heuristic training selection show the mean and standard deviation for 10 different runs of the algorithm.

	Random		Deterministic		Heuristic	
n	Training	Test	Training	Test	Training	Test
20	0.990±0.007	0.216±0.102	1.00	0.685	1.00±0.000	0.869±0.032
50	0.994±0.003	0.224±0.104	1.00	0.685	0.997±0.003	0.913±0.012
100	0.992±0.003	0.108±0.065	1.00	0.685	0.991±0.003	0.942±0.003
150	0.992±0.002	0.052±0.013	1.00	0.685	0.992±0.004	0.945±0.002
200	0.995±0.001	0.126±0.068	1.00	0.685	0.988±0.003	0.936±0.003
300	0.995±0.001	0.137±0.067	1.00	0.685	0.987±0.003	0.945±0.002
400	0.996±0.001	0.073±0.016	1.00	0.685	0.988±0.002	0.948±0.001
500	0.995±0.001	0.096±0.023	1.00	0.685	0.988±0.0025	0.949±0.002
1000	0.993±0.001	0.231±0.064	1.00	0.685	0.988±0.001	0.951±0.001

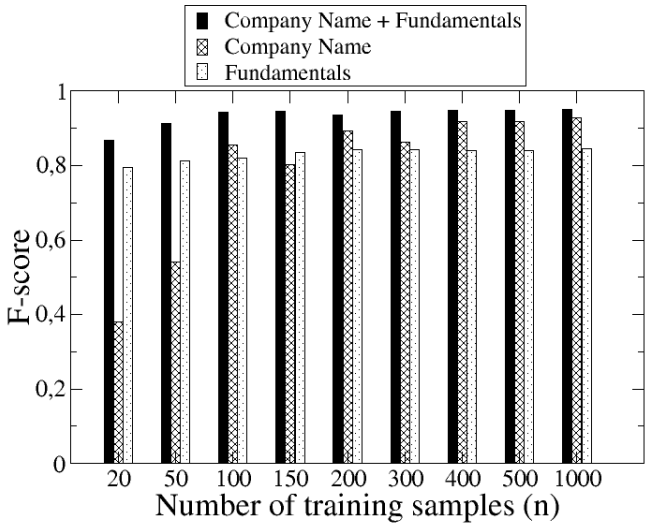


Figure 3 Comparison of the FASD's F-score in test for three different sets of attributes: the best four links for numeric attributes and the best link for string attributes found by FASD's schema matching (company name + fundamentals), the best string link (company name), and the best four numeric links (fundamentals). Results show the average for 10 different runs of the algorithm using the heuristic strategy for training selection.

Table 5 shows that FASD's company matching algorithm is able to effectively discover correspondences between companies. It yields an F-score of 0.94 when trained only with 100 positive and negative company links chosen according to our heuristic strategy and using the numeric and string attribute links found by FASD's schema matching method. In fact, the performance of the algorithm keeps almost constant for more than 100 training pairs, which means that increasing the labeling effort does not provide any advantage in terms of effectiveness. The other two scenarios considering only either string or numeric attributes (Figure 3) are not competitive enough even when they are provided with a large number of training samples. This result reveals the importance of taking into account heterogeneous information given the complex casuistry of company matching problems. Finally, the heuristic strategy for training set selection yielded the best results regardless of the set of attributes used. This is not a surprising result given that our heuristic approach focuses on those samples that are more difficult to differentiate and close to the decision boundary of the matching rule. On the other hand, the poor performances of the deterministic and random strategies are not surprising either. While the training set in the deterministic approach is likely to be dominated by positive examples, the negative training samples in the random strategy are unlikely to be *challenging* cases. In short, these training sets do not properly capture the complexity of the underlying problem.

4. Conclusions

The development of trading strategies using multimodal sources of information require a painful process of curating the data to avoid any sort of forward looking bias. We also need to have the data aligned in the proper format such that trading algorithms can be developed and trained. The data management problem eventually overtakes and hampers the model development process. Thus, this paper deals with a challenging problem for machine learning, databases, and finance communities consisting of discovering correspondences between companies in heterogeneous financial databases. The company matching problem is mainly characterized by the lack of a common company's identifier across databases originated at different times without a common standard that evolved in tandem with computer technological breakthroughs. Finance databases are treacherous and the curation of the data is time consuming. There is difficulty of finding correspondences between heterogeneous attributes, which requires the application of a schema matching algorithm. Moreover, there exist several records per company, which leads to reformulate traditional matching approaches commonly oriented to finding one-to-one correspondences. The challenge of having a common framework to make comparisons is hard because every database matching problem leads to a different subproblem. The finance databases shown here are characteristic of a generic situation not previously addressed in the literature in which 1/ time scales of the reported data are different, 2/ attributes may be conceptually similar but they are linearly or nonlinearly scaled and shifted, 3/ there are strings and numbers, 4/ unique key identifiers in each database are different, 5/ companies with different names are actually the same in both databases, 6/ companies with similar identifiers are actually different.

To solve the company matching problem, we propose a two-stage algorithm that requires no expert knowledge and low

human labeling effort. The first step of the algorithm provides a solution to the schema matching problem based on the computation of the Kullback-Leibler divergence between attributes. The algorithm is completely unsupervised and admits string and numeric attributes. The second stage is responsible for the company matching task, which is carried out in a supervised manner. FASD's company matching algorithm defines an auto-configurable company distance as a function of a set of parameters, and it lets an optimization algorithm to discover the *optimal* configuration according to the attribute links found in the first stage. Our company distance is capable of dealing with heterogeneous types of data (numeric and string) and entities (companies) defined by a ``time series'' of records with a time stamp associated. FASD's company distance is also able to automatically find a good combination of edit-based and token-based metrics, choose among different edit-based metrics, assign different weights (importance) to each attribute, and establish the optimal threshold that defines the matching rule. Experimental results on matching companies from Compustat/CRSP and I/B/E/S databases by considering different sets of attributes and training set selection strategies show that FASD is able to successfully discover correspondences between companies. FASD yields an F-score of 0.94 when a hundred positive and negative company links are intelligently chosen for training, and numeric and string attributes are considered.

Acknowledgements

This research is based upon work supported in part by the Office of the Director of National Intelligence (ODNI), Intelligence Advanced Research Projects Activity (IARPA), via the Federal Bureau of Investigations, Finance Division. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of ODNI, IARPA, or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright annotation thereon.

References

Baeza-Yates, R., Ribeiro-Neto, B. et al. (1999). *Modern information retrieval*. ACM press New York.

Bernstein, P. A., Madhavan, J., & Rahm, E. (2011). Generic schema matching, ten years later. *Proceedings of the VLDB Endowment*, 695-701.

Bilenko, M., Mooney, R., Cohen, W., Ravikumar, P., & Fienberg, S. (2003). Adaptive name matching in information integration. *IEEE Intelligent Systems, 18*(5), 16-23. http://dx.doi.org/10.1109/MIS.2003.1234765

Camacho, D., Huerta, R., & Elkan, C. (2008). *An Evolutionary Hybrid Distance for Duplicate String Matching.* Technical report, Universidad Autonoma de Madrid. Retrieved from http://arantxa.ii.uam.es/~dcamacho/StringDistance/hybrid-distance.pdf

Cohen, W., Ravikumar, P., & Fienberg, S. (2003). A comparison of string metrics for matching names and records. *KDD Workshop on Data Cleaning and Object Consolidation*, *3*, 73-78.

Commission, U. S. (n.d.). *CUSIP Number*. Retrieved from http://www.sec.gov/answers/cusip.htm

de Carvalho, M. G., Laender, A. H., Goncalves, M. A., & da Silva, A. S. (2012). A genetic programming approach to record deduplication. *IEEE Transactions on Knowledge and Data Engineering, 24*(3), 399-412. http://dx.doi.org/10.1109/TKDE.2010.234

Doan, A., Domingos, P., & Halevy, A. Y. (2001). Reconciling schemas of disparate data sources: A machine-learning approach. *ACM Sigmod Record*, *30*, 509-520. http://dx.doi.org/10.1145/375663.375731

Duda, R. O., Hart, P. E., & Stork, D. G. (2012). *Pattern classification*. John Wiley & Sons.

Elmagarmid, A. K., Ipeirotis, P. G., & Verykios, V. S. (2007). Duplicate record detection: A survey. *IEEE Transactions on Knowledge and Data Engineering, 19*(1), 1-16. http://dx.doi.org/10.1109/TKDE.2007.250581

Gal, A., & Shvaiko, P. (2009). Advances in ontology matching. In *Lecture Notes in Computer Science*, 176-198. Springer. http://dx.doi.org/10.1007/978-3-540-89784-2_6

Holland, J. H. (1975). *Adaptation in natural and artificial systems: An introductory analysis with applications to biology, control, and artificial intelligence.* U Michigan Press. Retrieved from https://mitpress.mit.edu/books/adaptation-natural-and-artificial-systems

Huerta, R., Elkan, C., & Corbacho, F. (2013). Nonlinear Support Vector Machines Can Systematically Identify Stocks with High and Low Future Returns. *Algorithmic Finance, 2*, 1-45. http://dx.doi.org/10.2139/ssrn.1930709

Isele, R., & Bizer, C. (2012). Learning expressive linkage rules using genetic programming. *Proceedings of the VLDB Endowment, 5*(11), 1638-1649. http://dx.doi.org/10.14778/2350229.2350276

Jaiswal, A., Miller, D. J., & Mitra, P. (2013). Schema Matching and Embedded Value Mapping for Databases with

Opaque Column Names and Mixed Continuous and Discrete-valued Data Fields. *ACM Trans. Database Syst., 38*(1), 1-34. http://dx.doi.org/10.1145/2445583.2445585

Jaiswal, A., Miller, D., & Mitra, P. (2010). Uninterpreted Schema Matching with Embedded Value Mapping under Opaque Column Names and Data Values. *IEEE Transactions on Knowledge and Data Engineering, 22*(2), 291-304. http://dx.doi.org/10.1109/TKDE.2009.69

Jaro, M. A. (1989). Advances in record-linkage methodology as applied to matching the 1985 census of Tampa, Florida. *Journal of the American Statistical Association, 84*(406), 414-420. http://dx.doi.org/10.1080/01621459.1989.10478785

Kang, J., & Naughton, J. F. (2008). Schema matching using interattribute dependencies. *IEEE Transactions on Knowledge and Data Engineering, 20*(10), 1393-1407. http://dx.doi.org/10.1109/TKDE.2008.100

Kim, K.-j., & Han, I. (2000). Genetic algorithms approach to feature discretization in artificial neural networks for the prediction of stock price index. *Expert systems with applications, 19*(2), 125-132. http://dx.doi.org/10.1016/S0957-4174(00)00027-0

Köpcke, H., & Rahm, E. (2008). Training selection for tuning entity matching. *QDB/MUD*, 3-12.

Köpcke, H., & Rahm, E. (2010). Frameworks for entity matching: A comparison. *Data & Knowledge Engineering, 69*(2), 197-210. http://dx.doi.org/10.1016/j.datak.2009.10.003

Köpcke, H., Thor, A., & Rahm, E. (2010). Evaluation of entity resolution approaches on real-world match problems. *Proceedings of the VLDB Endowment, 3*, 484-493. http://dx.doi.org/10.14778/1920841.1920904

Levenshtein, V. I. (1966). Binary codes capable of correcting deletions, insertions, and reversals. *Soviet physics doklady*, 707-710.

Levine, D. (1996). Users guide to the PGAPack parallel genetic algorithm library. *Argonne National Laboratory*. http://dx.doi.org/10.2172/366458

Liu, H., Dou, D., & Wang, H. (2012). Breaking the Deadlock: Simultaneously Discovering Attribute Matching and Cluster Matching with Multi-Objective Metaheuristics. *Journal on data semantics, 1*(2), 133-145. http://dx.doi.org/10.1007/s13740-012-0010-0

Monge, A. E., & Elkan, C. (1997). Efficient domain-independent detection of approximately duplicate database records. Retrieved from http://cseweb.ucsd.edu/~elkan/approxdup.pdf

Monge, A. E., Elkan, C. et al. (1996). The Field Matching Problem: Algorithms and Applications. *KDD*, 267-270. Retrieved from https://www.aaai.org/Papers/KDD/1996/KDD96-044.pdf

Moussawi, R. (2006). *Linking I/B/E/S and Compustat Data.* Wharton Research Data Services. Web.

Rahm, E., & Bernstein, P. A. (2001). A survey of approaches to automatic schema matching. *The VLDB Journal, 10*(4), 334-350. http://dx.doi.org/10.1007/s007780100057

Sewell, M. (2010). *The Application of Intelligent Systems to Financial Time Series Analysis.* Department of Computer Science, University College London, University of London.

Shvaiko, P. a. (2005). A survey of schema-based matching approaches. *Journal on Data Semantics IV*, 146-171. http://dx.doi.org/10.1007/11603412_5

Winkler, W. E. (1999). *The state of record linkage and current research problems.* Retrieved from https://www.census.gov/srd/papers/pdf/rr99-04.pdf

Zhao, H. (2010). Matching Attributes across Overlapping Heterogeneous Data Sources Using Mutual Information. *Journal of Database Management (JDM), 21*(4), 91-110. http://dx.doi.org/10.4018/jdm.2010100105

Zhao, H., & Ram, S. (2007). Combining schema and instance information for integrating heterogeneous data sources. *Data & Knowledge Engineering, 61*(2), 281-303. http://dx.doi.org/10.1016/j.datak.2006.06.004